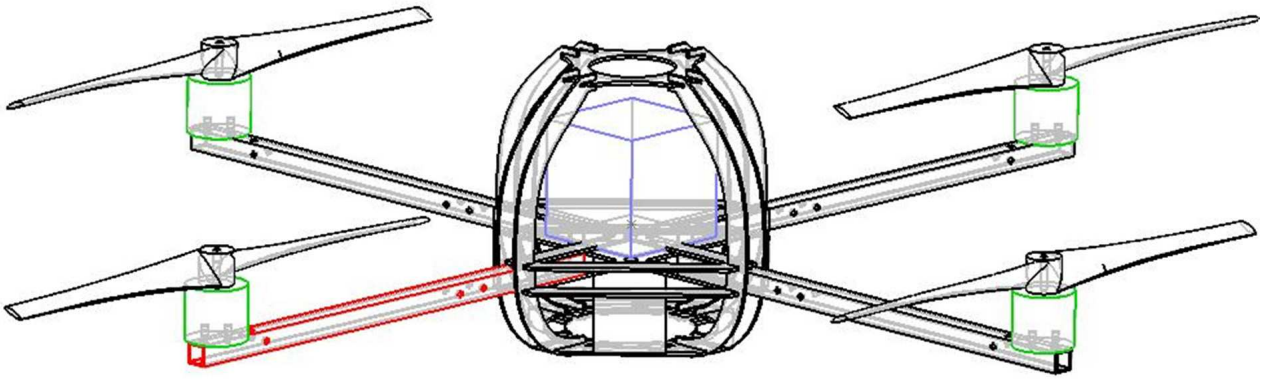


Controllo di un *Quadcopter*

Ch.mo Prof. Francesco Amato
Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria dell'Automazione

Univesità Federico II, Napoli

Giugno 2012

Giovanni Pugliese Carratelli
M58/30

Michele Del Duca
M58/18

Heavier than air flying machines are impossible
Lord Kelvin

Sommario

In questa tesina ci si propone di realizzare e stabilizzare l'assetto, di un particolare veivolo detto *Quadrirotore* o in inglese *QuadCopter*. Il Quadrirotore é un veivolo con una struttura portante su cui sono installati quattro motori elettrici con relative eliche che ne permettono il volo. Il quadrirotore è comandato da terra, e vista la sua intrinseca instabilità dispone di un sistema di controllo in grado di mantenerlo con un assetto livellato. In questa tesina verranno illustrate le varie fasi che si sono susseguite nello sviluppo di questo progetto, partendo da una prima fase, in cui si è costruito un modello matematico a sei gradi di libertá, una seconda fase che illustrerà delle simulazioni con alcune logiche di controllo ed infine una terza fase che mette in luce i vari aspetti e dettagli riguardanti la realizzazione fisica.

Indice

Capitolo 1

Quadrirotori

"Trovo, se questo strumento a vite sarà ben fatto, cioè fatto di tela lina, stopata i suoi pori con amido, e svoltata con prestezza, che detta vite si fa la femmina nell'aria e monterà in alto"

Leonardo Da Vinci, *Manoscritto B, foglio 83 v.*, 1483 – 1486

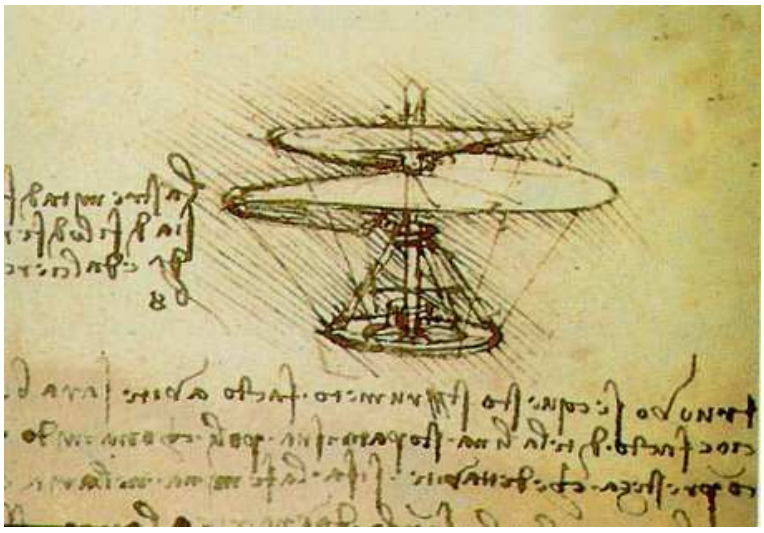


Figura 1.1: Progetto di elicottero di Leonardo

La prima idea di macchina che potesse *volare* è da attribuire al genio di Leonardo Da Vinci, il quale aveva intuito l'utilità di un congegno come la *vite aerea* per generare la *portanza* necessaria per mantenersi in volo. Dopo mezzo millennio di progresso, la tecnologia ha dato vita a velivoli dalle notevoli capacità, in grado di gestire autonomamente il volo (Unmanned Aerial Vehicle - UAV) e il cui campo applicativo è in continua espansione.

In questo primo capitolo vengono trattati gli aspetti principali che caratterizzano un quadricottero; si evidenziano le principali differenze rispetto a un elicottero convenzionale, per evidenziarne tutti i vantaggi che fanno di questa piattaforma uno degli strumenti più versatili e promettenti, ideale per diversi scenari applicativi. Inoltre, si descrivono le principali caratteristiche meccaniche di volo, utili per la comprensione del controllo del quadricottero.

1.1 Introduzione

Un quadricottero (dal greco *ptéron: ala*, "quattro ali") è un aerogiro sollevato e spinto da quattro rotori. Tale aeromobile multirotore così definito può essere anche classificato impropriamente come *elicottero*, pur essendo da quest'ultimo diverso per motivi tecnici, derivanti da evidenti differenze strutturali.

Le possibili applicazioni di un quadrirotore sono molteplici: dall'utilizzo di un quadrotor senza pilota per l'esplorazione di un territorio ostico, alla necessità di prestare un pronto intervento in uno scenario urbano dopo una catastrofe, che renderebbe il luogo inacessibile via terra e impraticabile per via aerea da qualunque altro mezzo di soccorso che non abbia le stesse caratteristiche di agilitá e destrezza del quadrirotore. L'attualitá dell'argomento trattato in questa tesina é testimoniata anche dalla recente commercializzazione del quadricottero-giocattolo *Parrot AR Drone* comandato mediante tecnologia wireless Wi-Fi da smartphone, questo é dotato di telecamera anteriore grandangolare usata per lo streaming delle immagini sul dispositivo di comando.

Per queste ragioni, in questo progetto si è scelto di studiare il funzionamento di un quadricottero, partendo dal suo modello matematico, per arrivare alla sintesi di diversi algoritmi di controllo, mirati a controllare il velivolo in volo, testati prima attraverso lo strumento di simulazione messo a disposizione dal Matlab/Simulink, poi applicati direttamente al modello di quadrirotore costruito interamente dagli autori di tale progetto.

1.2 Aspetti generali

Per meglio comprendere il funzionamento di un quadrirotore, é opportuno approfondire le principali caratteristiche tecniche di un elicottero da cui il quadrirotore in questione si differenzia. Il problema piú importante da risolvere per la stabilizzazione di un elicottero in fase di avanzamento é che le pale con componente tangenziale della velocità diretta nello stesso verso di avanzamento dell'elicottero si muovono più velocemente delle altre, generando così una portanza maggiore, tendendo quindi a ribaltare l'elicottero. Questo problema ha trovato soluzione con una grande innovazione nel rotore, inventata dall'ingegnere francese Ètienne Oehmichen, che permette di variare l'inclinazione ("angolo di attacco") delle pale ad ogni giro, in modo da equilibrare la portanza delle pale che hanno velocità assoluta più elevata con quella delle pale più lente. Questa inclinazione è regolabile secondo le necessità d'impiego attraverso un controllo detto "collettivo", con comando a *cloche*. A differenza di quanto appena detto, per un quadrottero utilizza pale a "passo fisso", il cui angolo di attacco non varia durante la rotazione.

Inoltre, ulteriore semplificazione tecnica e strutturale a vantaggio del quadrirotore è l'assenza del rotore di coda, invece presente nell'elicottero, le cui pale girano sul piano verticale per bilanciare la rotazione orizzontale di quelle del rotore principale. Il rotore di coda è un'"anticoppia" che si oppone alla coppia del rotore principale che determinerebbe una rotazione della fusoliera in senso opposto alla rotazione del rotore (legge di conservazione del momento angolare).

Un ulteriore vantaggio di un quadrirotore rispetto ad un elicottero é il minore diametro dei singoli rotori rispetto a quello necessario per un elicottero convenzionale, che consente di

imagazzinare minore energia cinetica: questo rende piú sicuro un quadrirotore sia in caso di incidente, che nell'utilizzo in ambienti *indoor*. Altri vantaggi, non di secondaria importanza, sono la simmetria del design che consente una centralizzazione dei sistemi di controllo e del payload: questa caratteristica agevola anche il sistema di controllo perché, anche con carichi diversi, è abbastanza semplice mantenere il baricentro nella stessa posizione; inoltre, ogni rotore contribuisce al raggiungimento della portanza richiesta, fornendo una quantità di spinta maggiore rispetto ad un elicottero convenzionale consentendo di portare payload e piattaforme computazionali più pesanti.

Infine, c'è da precisare che il quadricottero è un velivolo che rientra in una più ampia famiglia dei multirotori, ovvero aeromobili caratterizzati da più sistemi rotore-pala, che comprende al suo interno il tricottero, l'esacottero e l'octocottero (quest'ultimo particolarmente adatto a payload elevati). Dietro tutti questi vantaggi, c'è l'unico handicap di avere una bassa velocità di crociera.

Esistono due generazioni di modelli di quadrotor. La prima generazione fu pregettata per trasportare passeggeri. Tuttavia, è la generazione più recente ad aver riscosso il successo, essendo stata progettata per volare senza pilota a bordo (Unmanned Aerial Vehicle - UAV). Questi velivoli utilizzano un sistema di controllo e dei sensori elettronici per stabilizzarsi ed è su questa tipologia che si porrà l'attenzione in questa tesi.

> *"L'elicottero è una macchina volante e straordinaria. ...I volatili, ad esempio, hanno suggerito all'uomo l'aeroplano. l'elica da cui è nato l'elicottero può ritenersi, come la ruota, una pura creazione dell'ingegno umano. È una macchina straordinaria per le sue prestazioni di volo: è capace infatti non soltanto di atterrare e decollare verticalmente, e di traslare con moto rettilineo, ma di librarsi, di galleggiare* sur place *in alta quota come a pochi palmi dal suolo; è capace di ruotare in aria, su se stesso, di oscillare con moto pendolare, di volare di lato e all'indietro. Nemmeno gli eroi e i semi-dei della mitologia erano in grado di compiere tante prodezze"*
>
> Igino Mencarelli, *I padri dell'ala rotante, Aeronautica Militare Ufficio Storico*, 1969

1.3 Caratteristiche di volo

Da tutto ciò che è stato detto finora, si comprende che l'utilizzo di un aeromobile a quattro rotori rappresenta un'ottima soluzione per garantire facilità nel controllo e una forte stabilità, grazie alla disposizione a croce dei due assi sui quali sono disposti i quattro rotori.

Il quadriritore è un sistema sottoattuato poiché ha sei gradi di libertà: *beccheggio* (oscillazione del velicolo attorno ad un asse trasversale), *imbardata* (oscillazione del velicolo attorno all'asse verticale passante per il baricentro), *rollio* (oscillazione del velicolo attorno al proprio asse longitudinale), x (movimento nella direzione frontale del veicolo), y (movimento verso il lato sinistro del veicolo) e z (altitudine), ma è controllato utilizzando solo quattro attuatori.

Rotori opposti ruotano nello stesso verso. In particolare, quando tutti i rotori ruotano con la stessa velocità angolare, con i rotori 1 e 3 rotanti in senso orario ed i rotori 2 e 4 (vedi Figura 1.2), l'accelerazione angolare attorno all'asse di imbardata è nulla, il che implica che il rotore di coda presente negli elicotteri convenzionali non è necessario. L'imbardata viene provocata da una discrepanza nel bilanciamento del momento torcente aerodinamico, ovvero controbilanciando opportunamente i comandi di spinta tra le coppie di eliche che ruotano in senso opposto.

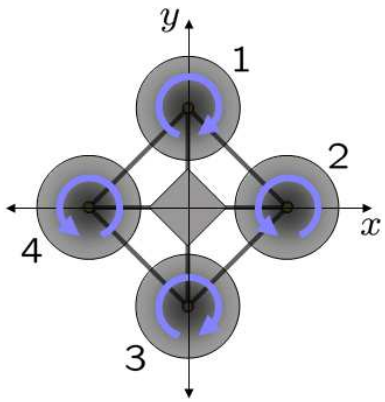
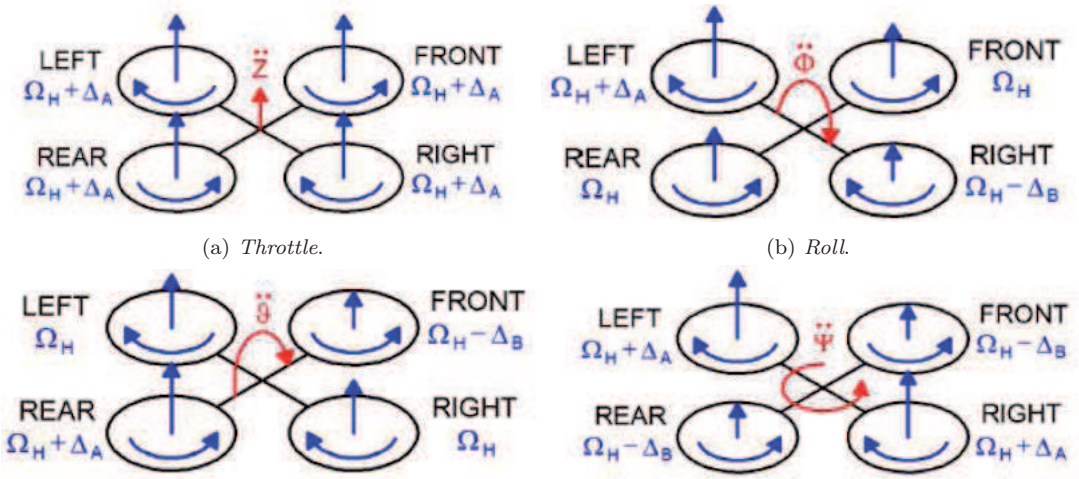


Figura 1.2: Schema di un quadricottero: I rotori *uno* e *tre* ruotano in senso orario, mentre i rotori *due* e *quattro* nella direzione opposta, provocando coppie con verso opposto per il controllo.

Accelerazioni angolari attorno agli assi di rollio e di beccheggio possono essere effettuate separatamente senza incidere sull'asse di imbardata. Ogni coppia di eliche disposta, che ruota quindi nello stesso verso, controlla la rotazione del velivolo lungo un asse, di rollio o di beccheggio. Aumentare la spinta di un rotore riducendo quella dell'altro permette di mantenere il bilancio di coppia necessario per la stabilità dell'imbardata e al tempo stesso produce una coppia netta attorno all'asse di rollio o di beccheggio. Per questo motivo si utilizzano rotori con pale a passo fisso per dirigere il velivolo lungo tutte le direzioni, a differenza di quanto avviene per un elicottero convenzionale, in cui è necessario disporre pale il cui angolo di attacco varia durante a rotazione, come già discusso nel precedente paragrafo.

Un'accelerazione di traslazione può essere raggiunta mantenendo un angolo di beccheggio o rollio diverso da zero.



(a) *Throttle.* (b) *Roll.*

(c) *Pitch.* (d) *Yaw.*

Figura 1.3: Convenzione movimenti modello a 4 rotori.

Capitolo 2

Modello matematico di un quadrirotore

Il primo passo prima di scegliere ed implementare le strategie di controllo, e dunque passare alla simulazione, è quello di modellare adeguatamente le dinamiche del sistema. In questa fase vengono illustrate le tecniche e le equazioni usate per modellare un quadrirotore nel suo complesso e dunque e vengono fissate le basi matematiche per la descrizione di un quardirotore in un ambiente di simulazione.

2.1 Dinamiche del sistema quadrirotore

Prima di poter descrivere le equazioni che governano un velivolo, come il quardirotore, è necessario introdurre le coordinate di riferimento in cui ci propone di descrivere l'assetto e la posizione. Nel caso del quadrirotore è possibile utilizzare due sistemi di riferimento, uno fisso, ed uno mobile solidale al telaio [1] le cui dinamiche possono essere espresse rispetto alla terna fissa. La terna fissa, detta anche inerziale o terna mondo, è una terna dove si può considerare valida la prima legge di Newton[2]. Nel sviluppo di questo modello è stato scelto di usare com terna fissa un terna chiamata in letteratura *NED* (North-East-Down), O_{NED} che come si evince dall'immagine che segue ha i versori gli assi rivolti verso Nord Est e verso il centro della terra.

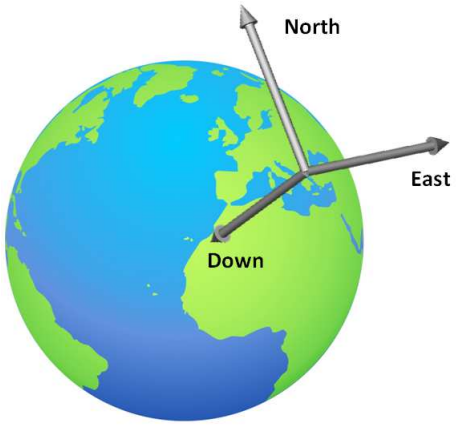


Figura 2.1: Il sistema di riferimento NED

[1] chiamato anche con terminologia inglese *frame*

[2] Un oggetto risluterà fermo a meno che non sottoposto ad una forza; un oggetto non varia la propria velocità (accelerazione) se non sottoposto all'azione di una forza

La terna mobile a cui si è fatto riferimento prima è invece solidale con il baricentro del quadcopter, ed è in letteratura nota come terna O_{ABC} dove ABC sta per *Aircraft Body Center*. L'immagine che segue mette in luce le due terne ed i rispettivi versori che le formano.

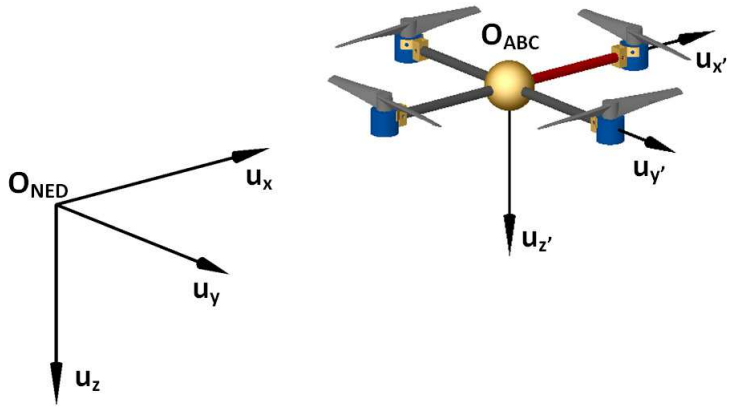


Figura 2.2: I sistemi di riferimento ABC e NED

Nella teoria del controllo le dinamiche del sistema modellato sono espresse tipicamente attraverso lo *stato* del sistema che nel caso del quadrirotore corrispondono a 12 equazioni che ne descrivono l'assetto e la posizione a sei gradi di libertà. Le variabili di interesse sono quindi 6 per il sistema NED ed altre 6 per il sistema ABC. In particolare definiamo come V_B e ω_B il vettore di velocità lineari e angolari del veivolo nella terna O_{ABC} con le seguenti variabili di stato:

$$V_B = [u\ v\ w]^T \qquad \omega_B = [p\ q\ r]^T \tag{2.1}$$

Per il sistema NED invece consideriamo le posizioni lineari:

$$\Gamma_{NED} = [x\ y\ z]^T \tag{2.2}$$

e gli angoli di roll pitch e yaw

$$\Theta_{NED} = [\phi,\ \theta,\ \psi]^T \tag{2.3}$$

Per procedere alla stesura delle equazioni in forma di stato a riguardo della terna NED è necessario l'uso di una matrice di trasformazione che permette di descrivere, componente per componente l'orientamento del frame mobile rispetto al fisso. La matrice di trasformazione complessiva sarà composta dal prodotto di 3 matrici di rotazione rispetto agli assi di rotazione $\phi,\ \theta,\ \psi$ ovvero le matrici che seguono:

$$R_\phi^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{bmatrix}, R_\theta^T = \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix}, R_\psi^T = \begin{bmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

La matrice risultante dal prodotto delle tre matrici appena definite è la matrice R^3 seguente:

$$R = \begin{bmatrix} cos\theta cos\psi & cos\theta sin\psi & -sin\theta \\ sin\phi sin\theta cos\psi - cos\phi sin\psi & cos\phi cos\psi + sin\phi sin\theta sin\psi & sin\phi cos\theta \\ cos\phi sin\theta cos\psi + sin\phi sin\psi & sin\theta cos\phi sin\psi - sin\phi cos\psi & cos\theta cos\phi \end{bmatrix} \tag{2.5}$$

[3]Anche nota come Direct Cosine Matrix

pertanto per i termini lineari si ha:

$$\dot{\Gamma}_{NED} = R \cdot V_B \tag{2.6}$$

e allo stesso modo in termini angolari si ha un atra matrice T per le trasformazioni angolari:

$$\dot{\Theta}_{NED} = T \cdot \omega_B \tag{2.7}$$

si osservi che la matrice T[4] per la trasformazioni angolari è la seguente:

$$T = \begin{bmatrix} 1 & sin\phi tang(\theta) & -sin\theta \\ 0 & cos\phi & -sin\phi \\ 0 & \frac{sin\phi}{sin\theta} & \frac{cos\phi}{cos\theta} \end{bmatrix} \tag{2.8}$$

Definite le trasformazioni che permettono di esprimere l'assetto del frame nella terna inerziale, è possibile applicare la seconda legge di Newton[5] facendo due ipotesi:

- L'origine della terna ABC corrisponde al CDM del quadricottero
- Gli assi della terna ABC corrispondono con assi principali di inerzia

Queste ipotesi permettono di scrivere un tensore di inerzia diagonale e di annullare i prodotti centrifughi e ottenere quindi un tensore diagonale. Se applichiamo quindi le leggi cardinali della dinamica si ha:

$$m\ddot{\Gamma}_{NED} = \sum_{i=0}^{n} F_i = F_{NED} \tag{2.9}$$

e in termini angolari:

$$I\ddot{\Theta}_{NED} = \sum_{i=0}^{n} \tau_i = \tau_{Ned} \tag{2.10}$$

a questo punto ricordandosi della trasformazione introdotta nella equazione 2.6 si può scrivere:

$$m\dot{\widehat{RV}}_B = RF_B \tag{2.11}$$

e quindi:

$$m(R\dot{V}_B + \dot{R}V_B) = RF_B \tag{2.12}$$

ricordandosi quindi che la velocità di un terna si può esprimere come velocità (lineare)della terna stessa più la sua rotazione si ha:

$$mR(\dot{V}_B + \omega_B \times V_B) = RF_B \tag{2.13}$$

$$m(\dot{V}_B + \omega_B \times V_B) = F_B \tag{2.14}$$

osserviamo che F_B sono le forzi agenti sul body frame del nostro quadrirotore e ω_B sono le velocità angolari nel sistema di riferimento del body.
Passando adesso a guardare cosa accade in termini angolari possiamo scrivere similmente a prima:

$$I\dot{\widehat{T\omega}}_B = T\tau_B \tag{2.15}$$

[4]si osservi che in realtà $T = E^{-1}$ in quanto la matrice E è più semplice da scrivere e lega i le velocità ω_B con la terna NED
[5]si trascureranno alcuni termini come il termine di Coriolis e quello aerodinamico

$$IT\dot{\omega_B} + T\omega_B \times (I\omega B) = T\tau_B \tag{2.16}$$

$$I\dot{\omega_B} + \omega_B \times (I\omega B) = \tau_B \tag{2.17}$$

dove τ_B è il risultante delle coppie nel riferimento ABC
Ora se sviluppiamo i prodotti vettoriali che abbiamo mostrato possiamo arrivare al seguente formalismo:

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix} = F_B \tag{2.18}$$

in termini angolari si ha:

$$\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = m \begin{bmatrix} \dot{p}I_x + qr(I_z - I_y) \\ \dot{q}I_y + pr(I_x - I_z) \\ \dot{r}I_z + pq(I_y - I_z) \end{bmatrix} = \tau_B \tag{2.19}$$

Osserviamo che quanto finora mostrato vale per un qualsiasi corpo rigido soggetto a delle forze ed a delle coppie.

2.1.1 Modello non lineare

Ora specifichiamo meglio cosa accade nel caso del quadrirotore andando a vedere quali sono le forze e le coppie agenti sul frame. Osserviamo che F_B può scomporsi in due pezzi il primo che dato dalla forza gravitazionale che chiamiamo F_B^g ed il secondo è chiamato con terminologia aeronautica *spinta*che chiamiamo F_B^T

$$F_B = F_B^g + F_B^T \tag{2.20}$$

il termine F_B^T è già espresso in termini del frame ABC mentre il campo gravitazionale terrestre deve essere riportato tramite la DCM. Si ha quindi infatti semplificando l'espressione della spinta a: $T = b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2)$

$$R \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} = m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix} \tag{2.21}$$

riorganizzando e sviluppando si ottiene:

$$\dot{u} = rv - qw - gsin\theta \tag{2.22}$$

$$\dot{v} = pw - ru - gcos\theta sin\phi \tag{2.23}$$

$$\dot{w} = qu - pv + gcos\phi cos\theta - \frac{T}{m} \tag{2.24}$$

In termini angolari sappiamo che per imprimere una coppia è sufficiente avere un differente velocità tra due motori sullo stesso asse. Questo avviene grazie alla inerzia dei motori ed in particolare le coppie sono calcolabili come:

$$M_x = lb(\Omega_2^2 - \Omega_4^2) \tag{2.25}$$

$$M_y = lb(\Omega_1^2 - \Omega_3^2) \tag{2.26}$$

$$M_z = d(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \tag{2.27}$$

In seguito vedremo come queste differenze di velocità permetteranno di scrivere le equazioni di controllo in modo particolarmente agevole.
In termini angolari allora le equazioni che governano il moto sono le seguenti:

$$\dot{p} = \frac{lb}{I_x}(\Omega_2^2 - \Omega_4^2) - qr\frac{Iz - Iy}{Ix} \tag{2.28}$$

$$\dot{q} = \frac{lb}{I_x}(\Omega_1^2 - \Omega_3^2) - pr\frac{Ix - Iz}{Iy} \tag{2.29}$$

$$\dot{r} = \frac{d}{I_z}(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \tag{2.30}$$

In questa trattazione non è stata modellato l'effetto giroscopico, certamente presente, ma viste le ridotte velocità in gioco lo abbiamo ritenuto trascurabile.
Ora ricordandosi delle trasformazioni introdotte all'inizio del paragrafo si può descrivere la dinamica *non lineare* del quadrirotore come segue:

$$\dot{\Gamma}_{NED} = R^{-1}V_B \tag{2.31}$$

$$\dot{u} = rv - qw - gsin\theta \tag{2.32}$$

$$\dot{v} = pw - ru - gcos\theta sin\phi \tag{2.33}$$

$$\dot{w} = qu - pv + gcos\phi cos\theta - \frac{T}{m} \tag{2.34}$$

$$\dot{\phi} = p + sin(\phi) + tan\theta r \tag{2.35}$$

$$\dot{\theta} = cos\theta q - sin\phi r \tag{2.36}$$

$$\dot{\psi} = \frac{sin\phi}{cos\theta}q + \frac{cos\phi}{cos\theta} \tag{2.37}$$

$$\dot{p} = \frac{lb}{I_x}(\Omega_2^2 - \Omega_4^2) - qr\frac{Iz - Iy}{Ix} \tag{2.38}$$

$$\dot{q} = \frac{lb}{I_x}(\Omega_1^2 - \Omega_3^2) - pr\frac{Ix - Iz}{Iy} \tag{2.39}$$

$$\dot{r} = \frac{d}{I_z}(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \tag{2.40}$$

2.1.2 Modello linearizzato

Il modello non lineare precedentemente introdotto non è utile ai fini della stesura di una logica di controllo lineare, e visti quindi anche visti i presupposti del corso è stato linearizzato attorno ad un punto di equilibrio. Sebbene calcolare punti di equilibrio con l'ausilio di strumenti di calcolo come Matlab risulti particolarmente semplice abbiamo comunque preferito procedere operando prima a calcolando i punti a mano e successivamente calcolando i punti numericamente.
I punti di equilibrio sono triviali per il quadrirotore, intuitivamente infatti una possibile posizione di equilibrio è quello di un *assetto livellato* ma ovviamente con una altitudine diversa da 0. Meno triviale è il calcolo dell' *ingresso di equilibrio*, tale ingresso di equilibrio porta il sistema

nella posizione di *hoovering* voluta; il calcolo di tale posizione sarà effettuato numericamente sulla base di alcune considerazioni sulla spinta prodotta dall'accoppiamento elica motore.
Pertanto nella seguente tabella sono mostrati i valori dell' equilibrio per lo stato [chiamato da adesso in poi X_0]: Per quanto riguarda l'ingresso di equilibrio questo è espresso in termini di

$x_0 = 0$	$y_0 = 0$	$z_0 = 0$
$u_0 = 0$	$v_0 = 0$	$w_0 = 0$
$\phi_0 = 0$	$\theta_0 = 0$	$\psi_0 = 0$
$p_0 = 0$	$q_0 = 0$	$q_0 = 0$

Tabella 2.1: Tabella punti di equilibrio dello stato X_0

velocità angolari ed è mostrato nella seguente tabella[indicato da adesso in poi come Ω_0]:

$\Omega_{1,0}$	$\Omega_{2,0}$	$\Omega_{3,0}$	$\Omega_{4,0}$
323	323	323	323

Tabella 2.2: Tabella punti di equilibrio dello stato X_0

Ed allora calcolando la derivata della funzione dello stato e valutando questa nello stato di equilibrio X_0, calcolando poi la derivata della funzione di ingresso e valutando questa in Ω_0 si ottiene il seguente modello linearizzato nell' intorno dell' equilibrio.[6]

$$\dot{x} = u \tag{2.41}$$

$$\dot{y} = v \tag{2.42}$$

$$\dot{z} = w \tag{2.43}$$

$$\dot{\phi} = p \tag{2.44}$$

$$\dot{\theta} = q \tag{2.45}$$

$$\dot{u} = -g\theta \tag{2.46}$$

$$\dot{v} = g\theta \tag{2.47}$$

$$\dot{w} = -2 * \Omega_0 \frac{b}{m}(\Omega_1 + \Omega_3 - \Omega_2 - \Omega_4) \tag{2.48}$$

$$\dot{p} = 2\frac{lb}{I_x}\Omega_0(\Omega_2 - \Omega_4) \tag{2.49}$$

$$\dot{q} = 2\frac{lb}{I_y}\Omega_0(\Omega_1 - \Omega_3) \tag{2.50}$$

$$\dot{r} = 2\frac{k}{I_z}\Omega_0(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \tag{2.51}$$

[6]per semplicità di notazione si omette il Δ per indicare che è nell'intorno dello stato e dell'ingresso di equilibrio

Scrivendo il modello in termini di matrici A,B si arriva riorganizzando il vettore di stato come $[u, v, w, x, y, z, \phi, \theta, \psi, p, q, r]$ si ottengono le seguenti matrici:[7]

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -g & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & g & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.52}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -2\Omega_h \frac{b}{m} & -2\Omega_0 \frac{b}{m} & -2\Omega_0 \frac{b}{m} & -2\Omega_0 \frac{b}{m} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 2\frac{lb}{I_x}\Omega_0 & 0 & -2\frac{lb}{I_x}\Omega_0 \\ 2\frac{lb}{I_y}\Omega_0 & 0 & -2\frac{lb}{I_y}\Omega_0 & \\ 2\frac{k}{I_z}\Omega_0 & 2\frac{lb}{I_x}\Omega_0 & 2\frac{lb}{I_x}\Omega_0 & 2\frac{lb}{I_x}\Omega_0 \end{bmatrix} \tag{2.53}$$

Prima di mostrare come stati costruiti i modelli lineari e non per la simulazione è importante chiarire come la logica di controllo sia stata applicata in maniera un po' diversa dal modello che verrà presentato. Infatti per semplificare la costruzione dei controllori abbiamo pensato di di fare agire l'azione di controllo *non direttamente sulla velocità di ogni singolo motore* ma bensi sulla differenza di velocità che permette un variazione dello stato. successivamente è stato costruito tramite l'inversione di un matrice un oggetto chiamato *mixer* che dati ingresso i segnali di controllo in termini di differenza di velocità calcolasse in uscita le velocità da assegnare ad ogni singolo motore. Per comprendere meglio quanto esposto effettuiamo prima di tutto un cambio di variabile e definiamo quindi:

$$U_1 = (-\Omega_1 - \Omega_2 - \Omega_3 - \Omega_4) \tag{2.54}$$

$$U_2 = (\Omega_2 - \Omega_4) \tag{2.55}$$

$$U_3 = (\Omega_1 - \Omega_3) \tag{2.56}$$

[7] si osservi che la matrice C è un matrice identica e la matrice D è un matrice di zeri.

$$U_4 = (\Omega_1 + \Omega_2 - \Omega_3 - \Omega_4) \tag{2.57}$$

Effettuato il cambio di variabili possiamo quindi definire la seguente matrice:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \tag{2.58}$$

Ora volendo costruire un controllore con azione di controllo sul sistema in termini di U_1, U_2, U_3, U_4 invertendo la matrice appena mostrata è possibile risalire alla velocità dei motori. In particolare si ha infatti:

$$\begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} = \begin{bmatrix} -0.25 & 0 & 0.5 & 0.25 \\ -0.25 & 0.50 & 0 & -0.25 \\ -0.25 & 0 & -0.5 & 0.25 \\ -0.25 & -0.5 & 0 & -0.25 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} \tag{2.59}$$

2.1.3 Implementazione in Matlab

I modelli visti sono stati implementati in ambiente Simulink. In particolare il modello non lineare per efficienza di calcolo è stato costruito usando alcuni blocchi predefiniti di Simulink. Questo modello è stato usato per testare la validità del controllore costruito sul modello linearizzato illustrato precedentemente.

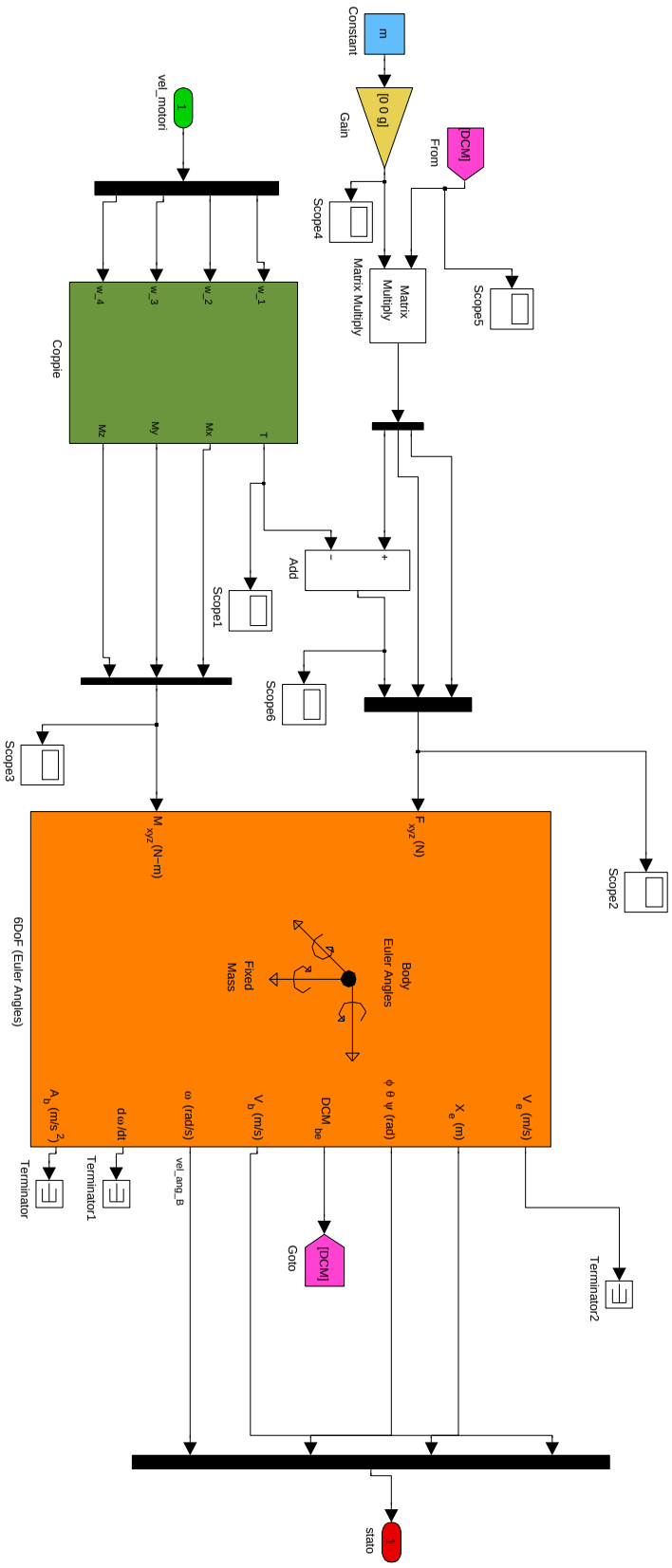


Figura 2.3: Modello non lineare

All'interno del blocco verde troviamo il modo in cui sono calcolate le forze e le coppie agenti su quadrirotore.

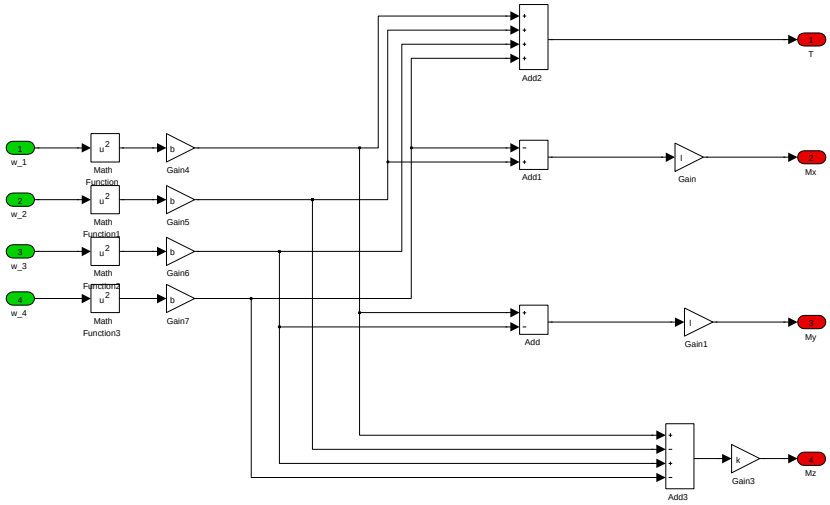


Figura 2.4: Modello non lineare

Alleghiamo per completezza anche il listato di cui si è fatto uso per una linearizzazione numerica e per caricare i parametri per le simulazioni.

```
Ix=0.0081; %kg/m2
Iy=Ix;
Iz=0.0162;
m=0.85;%kg
l=0.2; %(m) lungezza braccio da CDM
b=1.46*10^-5; % coefficiente di spinta
k=0.026;

g=9.81; % costante di gravità

%% Parametri motori
tau_mot=0.1; % costante di tempo dei motori

K_mot1=2.032;
K_mot2=1.90932;
K_mot3=2.131;
K_mot4=2.1121;
omega_0=323;
%% Linearizzazione
```

```
x0=zeros(12,1);
x0(6)=0.000001;
u0=323*ones(4,1);
[X,U,Y,DX]=TRIM('Modello_quad_6DOF',x0,u0,[],[1,2,3,4,5,6,7,8,9,10,11,12],
[A,B,C,D]=LINMOD('Modello_quad_6DOF',X,U);
omega_h=U(1);
```

2.2 Modello matematico degli attuatori

Al fine di avere un modello complessivo del quadrirotore, si è pensato di affinare la modellistica del sistema aggiungendo alla dinamiche del volo dell'*UAV* il modello matematico dei motori. Abbiamo suddiviso, per ragioni che verranno a breve spiegate la fase di di identificazione degli attuatori in due parti. Una prima dedita all'identificazione della cosi detta *lower level* dei motori, ed un seconda mirata a trovare la funzione di trasferimento dei motori stessi.

2.2.1 Identificazione funzione di trasferimento attuatori

Senza dover scendere nella specifica trattazione di un modello di un motore elettrico, tipicamente modellato come un sistema del second'ordine i cui due poli sono rispettivamente il polo delle dinamica elettrica e di quella meccanica (a tal proposito ci si può riferire in materia a testi come [1]), abbiamo pensato di modellare le dinamiche del sistema prendendo in considerazione quella più lenta, ovvero quella meccanica. Il trascurare le dinamiche elettriche permette quindi di costruire un modello che tenga conto solo delle dinamiche dominanti (quelle meccaniche) ed è pertanto rappresentabile come una funzione di trasferimento del prim'ordine del tipo:

$$G(s) = \frac{K}{(s\tau + 1)} \tag{2.60}$$

Sebbene sia quindi possibile intuire la strutta matematica (un sistema LTI del prim'ordine) generale degli attuatori, non abbiamo su questa informazioni di carattere numerico; risultano ignoti infatti i parametri K e τ. Conoscere tali parametri riveste un ruolo fondamentale non solo nella scelta del controllore, ma anche nelle simulazioni che vengono effettuate dove dinamiche lente o eventuali ritardi possono devono essere portati in conto. Per di più oltre ad un determinazione di quelli che sono i parametri K e τ dei singoli motori, non è ragionevole pensare che i 4 motori di bordo siano perfettamente uguali e quindi abbiano pertanto abbiamo pensato di affrontare una fase di *identificazione* dei parametri dei 4 motori. Osserviamo che sebbene possa essere un scelta discutibile quella di scegliere prima gli attuatori e poi identificarli, e dunque non procedendo a fare delle simulazioni per poi scegliere i motori successivamente, articoli e materiale a questo riguardo sono stati trovati in numero sufficiente per permette una scelta *prima* di andare a simulare il sistema.
Con riferimento ai motori scelti che verranno introdotti e descritti nel cap.5 abbiamo pensato di operare una fase di identificazione basata sulla conoscenza del segnale di comando dato ai motori (da noi generato e dunque noto) e sulla misura della velocità della rotazione delle pale

dell'elica installate sull'asse dei motori. Infatti la struttura matematica introdotta precedentemente si riferisce una funzione di trasferimento dove per ingresso è considerato il segnale di attuazione (che come verrà mostrato più avanti nel Cap. 5 è un segnale PWM a 50 Hz con duty cycle variabile) e per uscita è considerata la velocità angolare del motore espressa in $rads^{-1}$. Noto pertanto il segnale di ingresso al motore e noto il segnale di uscita è possibile costruire con alcuni strumenti elementari di identificazione un stima abbastanza veritiera dei parametri della funzione di trasferimento del processo.

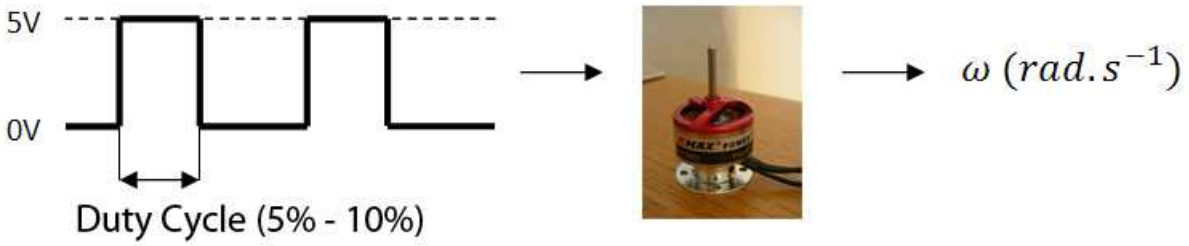
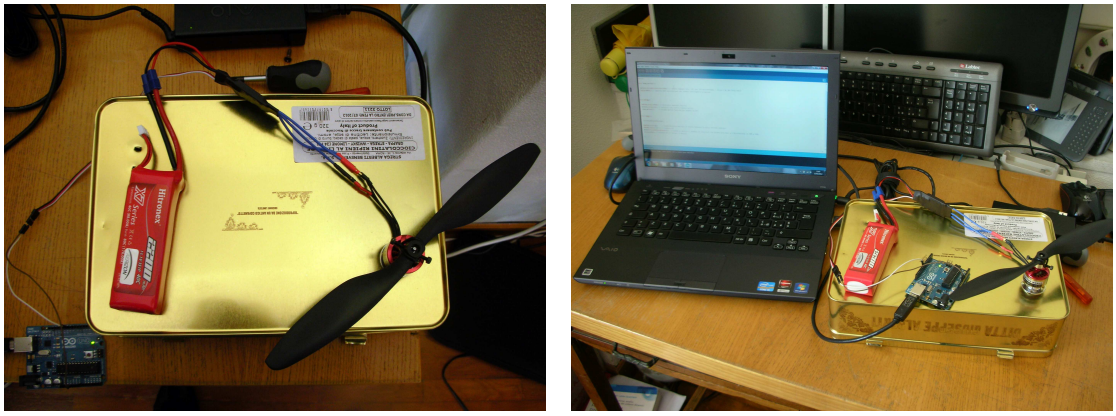


Figura 2.5: Idea alla base del processo di identificazione della funzione di trasferimento dei motori

Al fine di avere un misura della velocità angolare del motore abbiamo pensato di costruire un tachimetro (molto) rudimentale. Il tachimetro è stato costruito mediante l'uso di un microfono. Il moto delle pale delle eliche, infatti, per generare *portanza* e dunque spinta (ci si riferisca al paragrafo precedente a questo proposito) muovendosi genera sotto le pale stesse un fronte d'onda di pressione che muove la membrana del microfono e pertanto può con apposito strumento essere registrato.
Per gestire i segnali di controllo per il motore e effettuare le registrazioni abbiamo usato sia l'ambiento di lavoro Matlab che la MCU Arduino. Infatti per la generazione dei segnali di controllo dei motori abbiamo scritto un software (Arduino) che generasse un segnale PWM (50 Hz)con un assegnato duty cycle e per quanto riguarda la registrazione abbiamo usate alcune funzioni dell'ambiente Matlab.
Per quanto riguarda la costruzione del tachimetro abbiamo innanzitutto fissato ogni motore (uno per volta) con elica installata su di un scatola che lo tenesse fermo in posizione; la scatola utilizzata è una scatola in alluminio di cioccolatini. Come si evince dalla foto di Fig. 2.6.

Figura 2.6: Motore fissato alla base in alluminio per la fase di identificazione

Dopo aver fissato il motore sulla base di appoggio, lo abbiamo collegato alla E.S.C e l'abbiamo collegata alla MCU Arduino, a sua volta collegata al PC anche in questo caso la foto di Fig. 2.7 è esplicativa.



(a) (b)

Figura 2.7: Schema identificazione motori

Per registrare l'onda di pressione esercitata dalle pale abbiamo installato un microfono sotto le pale, aggiungendo però una seconda scatola sotto la precedente perché non siamo riusciti a far passare il microfono sotto l'elica senza cambiarne troppo la naturale posizione, come si evince dall figura 2.8

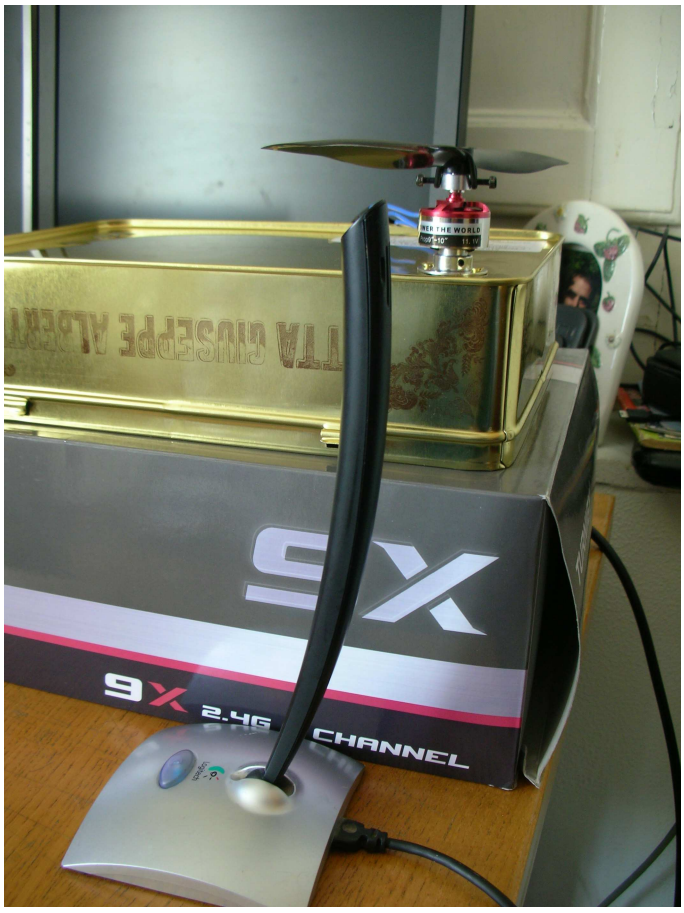


Figura 2.8: Installazione microfono nel provvisorio laboratorio

L'ambiente di lavoro su cui siamo andati a lavorare è stato pertanto quello di Fig. 2.9.

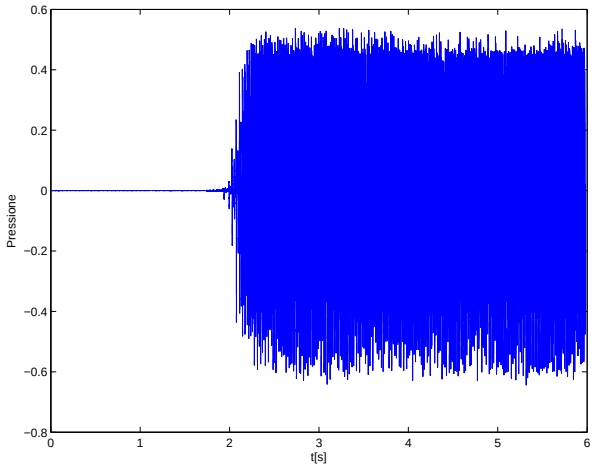
Figura 2.9: Ambiente di lavoro per l'identificazione dei motori

A mezzo di alcuni script abbiamo costruito una semplice funzione discreta avente per do-

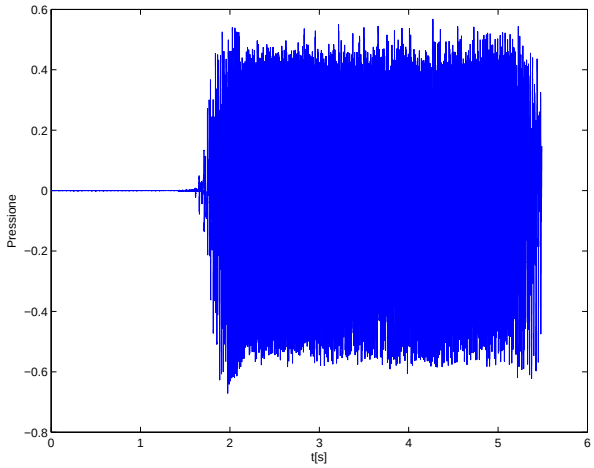
minio l'insieme discreto dei valori di duty cycle con passo pari a 100μs ed per condominio la velocità espressa in radianti a secondo. Su questa abbiamo calcolato il valore del guadagno K; per la costante di tempo abbiamo provveduto a fare la media delle costanti di tempo evinte empiricamente dalle singole prove a velocità fissata. Al fine di poter effettuare le prove per l'identificazione abbiamo utilizzato i due ambienti di sviluppo software offerti rispettivamente da Matlab e da Arduino. In particolare abbiamo scritto del codice in Matlab per effettuare la registrazione della onda di pressione proveniente dall'elica del motore, ed abbiamo anche scritto del codice per Arduino per generare i segnali di controllo da dare ai motori a più valori del Duty Cycle.
Il codice Matlab usato è stato semplice da scrivere e consiste nel campionare ad un frequenza di 44100Hz l'onda di pressione proveniente dal microfono. Al fine di effettuare queste misurazioni abbiamo fatto uso della funzione *audiorecorder()*, funzione questa che crea un oggetto nel workspace nel quale è possibile memorizzare un segnale audio a voluta frequenza di campionamento con il comando *record()*. Al termine della registrazione abbiamo archiviato in dei vettori le onde di pressione registrate.
Il lato Arduino invece è stato un po' più delicato da costruire e a tal proposito si allega il codice:

```
#include <Servo.h>

Servo motore;

void arma_motore(){

  motore.writeMicroseconds(800); // genera segnale alto (5V) per 800 microsecondi ( circa il 4% del Duty Cycle )
  delay(4000); // aspetta 4000 millisecondi per armare correttamente il motore

}

void setup()
{
  //Serial.begin(9600); // Definisci apertura seriale
  motore.attach(9); // definisci pin uscita segnale motore
  arma_motore(); // chiama questa funzione per armare le ESC.
  delay(2000); //margine
}

void loop(){

  motore.writeMicroseconds(1900); // manda al motore il segnale di controllo con durata alta pari a i us su PWM a 50Hz
}
```

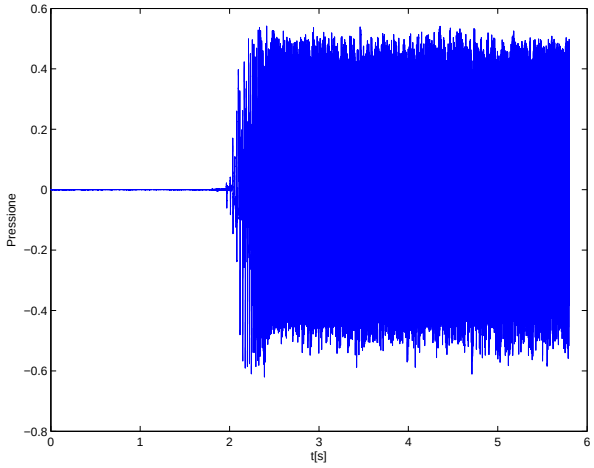
Per una singola prova quindi abbiamo in primo luogo fatto iniziare la registrazione e successivamente abbiamo dato il comando di avvio motori a mezzo del tasto di reset sull'Arduino. Alcuni esempi di registrazione che abbiamo ottenuto sono i seguenti:



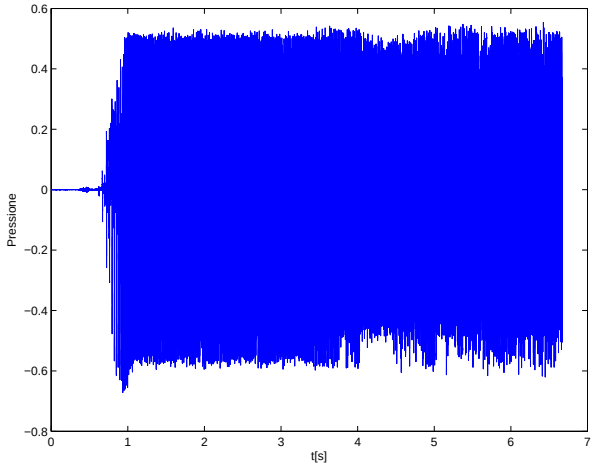
(a) Motore 1



(b) Motore 2



(c) Motore 3



(d) Motore 4

Dalle registrazioni ingrandendo lungo l'asse del tempo abbiamo misurato il periodo angolare (sebbene in alcune occasioni il segnale si sia presentato particolarmente rumoroso a causa di probabile vortici formati vicini al microfono). Il periodo angolare del motore è $2T_p$ ed è stato trovato dall'analisi del grafico.

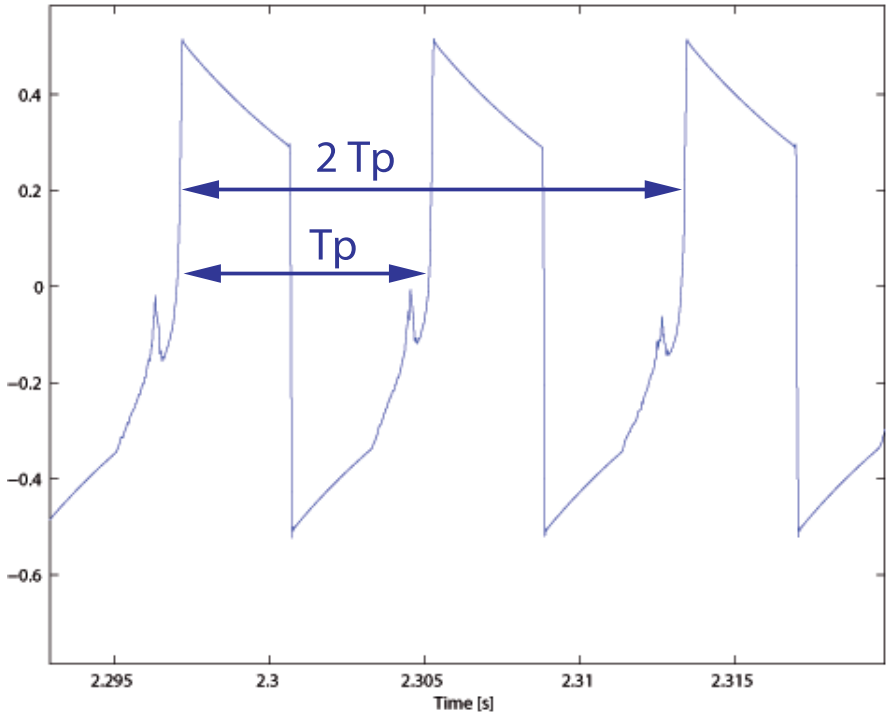


Figura 2.10: Zoom registrazione onda di pressione dall'elica dei motori, con ingresso un segnale con duty cycle di durata 1800μs e periodo misurato in uscita pari $2T_p = 0.017$

Dopo avere calcolato T_p è possibile calcolare la velocità angolare, associata ad un segnale di ingresso noto in quanto da noi generato, con la seguente formula:

$$\omega = \frac{\pi}{T_p} \tag{2.61}$$

A conferma di quanto abbiamo evinto dai grafici nel tempo una analisi spettrale mostra una componente principale esattamente ad una frequenza corrispondente a quella dell'inverso del periodo $2T_p$ misurato, cioè prossima a 120Hz. Si osservi inoltre che le componenti fuori dalla frequenza a 120Hz sono probabilmente dovute alle oscillazioni del rudimentale sistema di misura costruito e dal rumore generato dai probabili vortici.

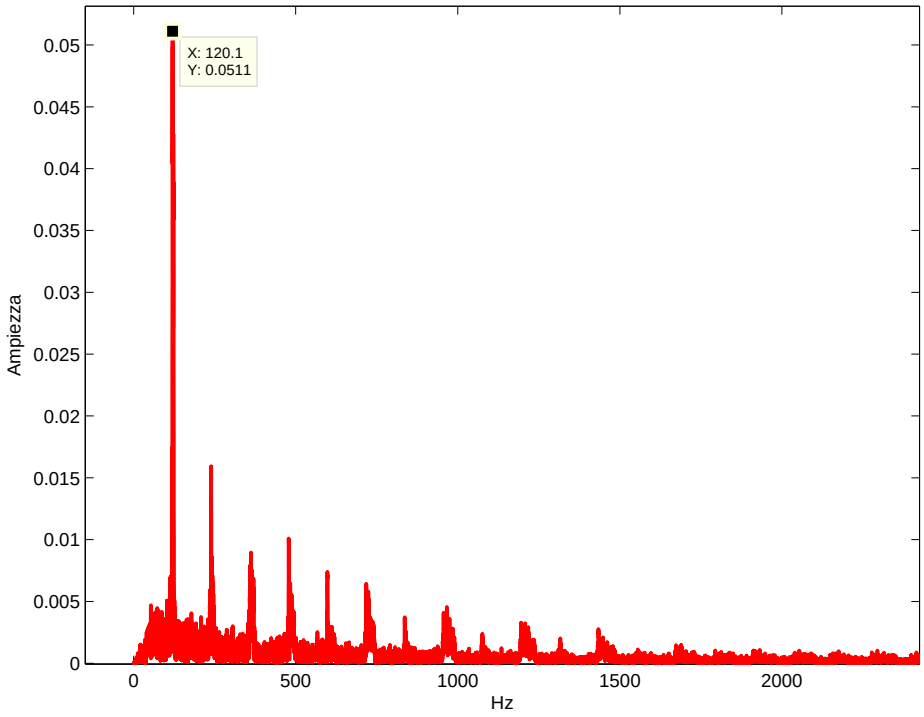


Figura 2.11: Spettro frequenziale dell'uscita del motore con ingresso PWM a 50Hz e duty cycle pari a 1800μs

Al termine delle varie prove abbiamo messo assieme i risultati ed abbiamo costruito per ogni motore un diagramma interpolato delle velocità registrate in funzione del duty cycle espresso in μs. Come si vede dai grafici il legame non è lineare ma è comunque possibile calcolare la K cercata prendendo la *differenza prima* valutata nel primo valore di duty cycle.

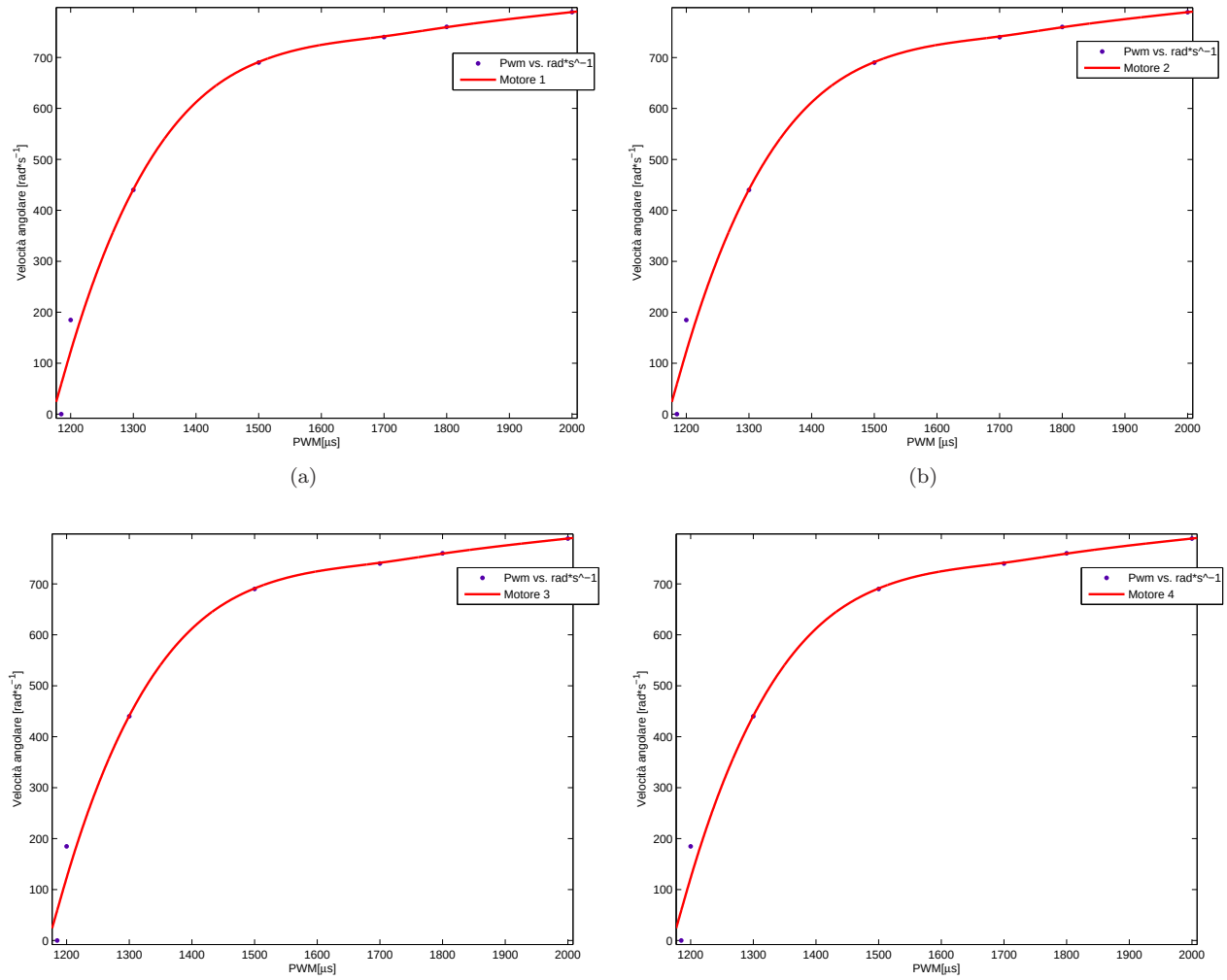


Figura 2.12: Interpolazione dati legame ingresso uscita motore per i 4 motori

I risultati ottenuti per i guadagni sono stati i seguenti:

Guadagno	Motore 1	Motore 2	Motore 3	Motore 4
$K_{1,2,3,4}$	2.00237	1.90321	2.012184	1.89313

Tabella 2.3: Tabella guadagni motori

Per il calcolo delle costanti di tempo è stata effettuata la media dei tempi di assestamento(diviso 4.6) per ogni singolo motore alle varie velocità. Vista la poca accuratezza dei dati è stato allora scelta i fissare $\tau = 0.1$ a tutti e 4 i motori. Le quattro funzioni di trasferimento, con riferimento alla precedente tabella, sono le seguenti:

$$G(s)_{1,2,3,4} = \frac{K_{1,2,3,4}}{(s\tau + 1)} \tag{2.62}$$

2.2.2 Identificazione lower level

Oltre ad identificare i parametri della funzione di trasferimento dei motori abbiamo anche avuto cura di identificare quella che è la *soglia minima di attivazione* di ogni singolo motore, paramento che poi ci è stato utile nella stesura del software di controllo. E' utile precisare che

le E.S.C (per maggiore chiarezza ci si riferisca alla Fig.5.10) aspettano un segnale PWM con frequenza fissata a 50 Hz e quindi periodo 20ms, e considerano la soglia bassa un duty cycle pari a circa il 5% e un soglia alta un dudty cycle prossimo al 10%. Al fine quindi di sapere quale sia il valore esatto della soglia di attivazione abbiamo effettuato delle prove con duty cycle crescente; tuttavia nel software scritto non abbiamo ragionato in termini percentuali di periodo bensi in termini di tempo (espresso in microsecondi). Con banali passaggi si ottengono i valori corrispondenti tra il duty cycle pari espresso in termini percentuali ed in termini temporali. Questa corrispondenza è riportata nella seguente tabella:

Duty Cycle	Micro secondi
5%	1000
10%	2000

Tabella 2.4: Legame Duty-Cycle percentuale/μs

Sulla base di questa conoscenza e attesa una partenza dei motori con valori percentuali prossimi al 5% abbiamo scritto un software in cui abbiamo fatto crescere linearmente il valore del duty cycle con un passo di 1μs a partire dal valore di 1050μs. Contemporaneamente all'invio dei segnali al motore è stato inviato on line tramite connessione USB (seriale) il valore espresso in micro secondi che veniva inviato ai motori. La valutazione dell'effettiva messa in movimento dei motori è stata semplicemente fatta visivamente ed il valore di attivazione riscontrato è risultato essere 1185μs, valore tutto sommato lontano dai valori del 5% dichiarato dal produttore a conferma di quanto possa essere importante una fase di identificazione.
Le prove sono state effettuate tramite MCU Aruino e non su piattaforma Ardunino Mini Pro usata poi per il controllo del modello di quadcopter. Alleghiamo il software utilizzato dovendo tutta via precisare che nella funzione setup è stato necessario *armare* i motori e le E.S.C (per migliori chiarimenti riferirsi al Capitolo 5).

Listing 2.1: Programma per identificazione soglia bassa motore

```
1
2  #include <Servo.h>
3
4  Servo motore;
5  int i;
6
7  void arma_motore(){
8
9    motore.writeMicroseconds(990); // genera segnale alto (5V) per 990 microsecondi ( circa il 4.99% del Duty Cycle)
10   delay(4000); // aspetta 4000 millisecondi per armare correttamente il motore
11
12 }
13
14 void setup()
15 {
16   Serial.begin(9600); // Definisci apertura seriale
17   motore.attach(9); // definisci pin uscita segnale motore
18   arma_motore(); // chiama questa funzione per armare le ESC.
19 }
20
```

```
21 void loop(){
22
23   for(i=1150;i<1300;i++){
24
25   Serial.print(i); // manda su porta seriale il valore espresso in micro secondi generato
26   Serial.print('\n'); // Sengale CR
27   motore.writeMicroseconds(i); // manda al motore il segnale di controllo con durata alta pari a i us su PWM a 50Hz
28   delay(1000); // aspetta che si azioni il motore
29   }
30 }
```

2.2.3 Modello Simunlink per gli attuatori

Il modello Simulink per gli attuatori è stato costruito come segue utilizzando anche la saturazione per modellare l'impossibilità di poter andare oltre un duty cycle del 20%.

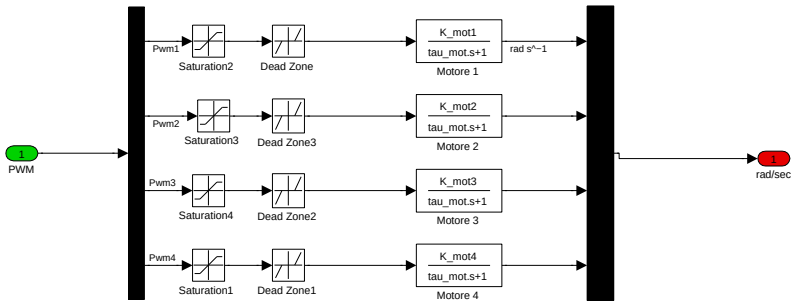


Figura 2.13: Modello Simulink per gli attuatori

2.3 Calcolo dei momenti di inerzia

Il calcolo dei momenti di inerzia è necessario al fine poter fare delle simulazioni accurate. Grazie alla particolare simmetria del quadrirotore il calcolo risulta particolarmente semplice a mezzo di alcune semplici formule di meccanica classica. E' chiaro che il calcolo non sarà precisissimo ma è tutta via sufficiente ad avere un stima di massima utile per la simulazione con l'auspicio che il controllore sia sufficientemente robusto a tollerare alcune variazioni parametriche di questo parametro.
Il calcolo viene effettuato considerando che il frame Abc è una terna centrale di inerzia e la matrice di inerzia sia diagonale in quanto i prodotti centrifughi si annullano. Al fine del calcolo è da considerare che la massa di ogni motore è pari $m_m = 0.048Kg$ ed il peso del quadrirotore

è prossimo a $m = 0.85Kg$. In più le distanza dei motori rispetto ai tre assi su cui giace la terna sono le seguenti: $l_x = l_y = 0.028, l_z = 0.026$; il braccio sotteso dai motori rispetto al centro di massa è pari ad $l_{CDM} = 0.28m$. E' possibile calcolare il momento di inerzia lungo x risolvendo il seguente sistema:

$$\begin{cases} I_{x_1} = I_{x_3} = \frac{1}{12}(l_y^2 + l_z^2) = 6.021810^{-6} Kgm^2 \\ I_{x_2} = I_{x_4} = \frac{1}{12}(l_y^2 + l_z^2) + m_m * l_{CDM} = 0.004 Kgm^2 \\ I_{11} = 2 * I_{x_1} + 2 * I_{x_2} = 0.0081 Kgm^2 \end{cases} \tag{2.63}$$

Stesso dicesi per il calcolo lungo gli assi y:

$$\begin{cases} I_{y_1} = I_{y_3} = \frac{1}{12}(l_x^2 + l_z^2) + m_m * l_{CDM} = 0.004 Kgm^2 \\ I_{y_2} = I_{y_4} = \frac{1}{12}(l_x^2 + l_z^2) = 6.021810^{-6} Kgm^2 \\ I_{22} = 2 * I_{y_1} + 2 * I_{y_2} = 0.0081 Kgm^2 \end{cases} \tag{2.64}$$

e per l'asse z:

$$\begin{cases} I_{z_1} = I_{z_2} = I_{z_3=} = I_{z_4} = \frac{1}{12}(l_y^2 + l_z^2) + m_m * l_{CDM} = 0.004 Kgm^2 \\ I_{33} = 4 * I_{z_1} = 0.0162 Kgm^2 \end{cases} \tag{2.65}$$

Si arriva pertanto al seguente tensore di inerzia:

$$I = \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix} = \begin{bmatrix} 0.0081 & 0 & 0 \\ 0 & 0.0081 & 0 \\ 0 & 0 & 0.0162 \end{bmatrix} \tag{2.66}$$

Capitolo 3

Strategie di controllo

Nel presente capitolo analizzeremo prima in linea generale le strategie di controllo finora studiate durante il corso di studi e in seguito approfondiremo in dettaglio ciascuna di esse, in relazione alla loro applicazione specifica sul modello di quadcopter oggetto di studio. In generale, Il problema del controllo consiste nel progettare un controllore C che, sulla base delle informazioni fornite da r(t) e y(t), sia in grado di generare un segnale u(t) che faccia evolvere il sistema S in modo da minimizzare lo scostamento tra y e r. Di seguito, si parte dall'analisi del controllo Proporzionale-Integrale-Derivativo basato sul rilevamento dell'uscita del sistema, per arrivare a tecniche di controllo più sofisticate basate sulla conoscenza dello stato del sistema (*full information*), quali *allocazione dei poli* (*pole placement*) e *LQR* (*Linear quadratic regulator*).

3.1 Controllori PID

Il controllo Proporzionale-Integrale-Derivativo, comunemente abbreviato come PID, è un sistema in retroazione negativa ampiamente impiegato nei sistemi di controllo. È il sistema di controllo in retroazione di gran lunga più comune nell'industria, in particolare nella versione PI (senza azione derivativa). La reazione all'errore può essere regolata, ciò rende questo sistema molto versatile. Il controllore acquisisce in ingresso un valore da un processo, e lo confronta con un valore di riferimento. La differenza, il cosiddetto segnale di errore, viene quindi usata per determinare il valore della variabile di uscita del controllore, che è la variabile manipolabile del processo. Il PID regola l'uscita in base a:

- il valore del segnale di errore (azione proporzionale);
- i valori passati del segnale di errore (azione integrale);
- quanto velocemente il segnale di errore varia (azione derivativa).

I controllori PID sono relativamente semplici da comprendere, installare, e tarare al confronto con più complessi algoritmi di controllo basati sulla teoria del controllo ottimo e del controllo robusto. Per questo motivo i controllori PID sono spesso sufficienti a controllare processi industriali anche complessi, ma la loro stessa semplicità risulta anche presente in una serie di limiti che è bene ricordare:

- non sono in grado di adattarsi a cambiamenti nei parametri del processo;
- non sono stabili, a causa della presenza dell'azione integrale (effetto Windup);
- alcune regole di taratura, come quelle di Ziegler-Nichols, reagiscono male in alcune condizioni;
- sono intrinsecamente monovariabili, non possono quindi essere usati in sistemi inerentemente multivariabili.

Alcune di queste limitazioni possono essere superate attraverso l'utilizzo di strategie alternative: si pensi, ad esempio, in riferimento all'adattamento ai cambiamenti parametrici del processo, al *gain scheduling* come soluzione empirica per rendere un controllore adattabile ai diversi punti di lavoro del sistema, ampiamente utilizzato nell'ambito delle applicazioni aeronautiche e automobilistiche. In particolare, l'idea che è alla base del suo sviluppo è la linearizzazione del sistema non lineare attorno a N punti di equilibrio, i cosiddetti *trim points*, e la successiva taratura dei parametri PID su tale rappresentazione lineare, valida solo localmente al valore della variabile di *scheduling* da controllare.
Relativamente alla limitatezza del controllore PID per il suo utilizzo in sistemi multivariabili, è possibile ricorrere ad un numero maggiore di controllori PID, equivalente al numero di variabili da controllare (è quello che faremo come prima metodologia di controllo per il quadricottero).
Il controllo PID è la tecnica di controllo più semplice ed economica da realizzare, nonchè la meno efficiente rispetto alle altre tecniche oggetto di studio durante il corso e che verranno analizzate nel seguito di questo capitolo, perchè si basa su un retroazione dell'uscita e non dello stato del sistema. La figura 3.1 seguente mostra un classico sistema di controllo in retroazione, dove il blocco di regolazione R può rappresentare la sintesi delle azioni *Proporzionali* (P) *Integrali* (I) e *Derivative* (D), presenti tutte insieme o in parte e che di seguito verranno singolarmante analizzate.

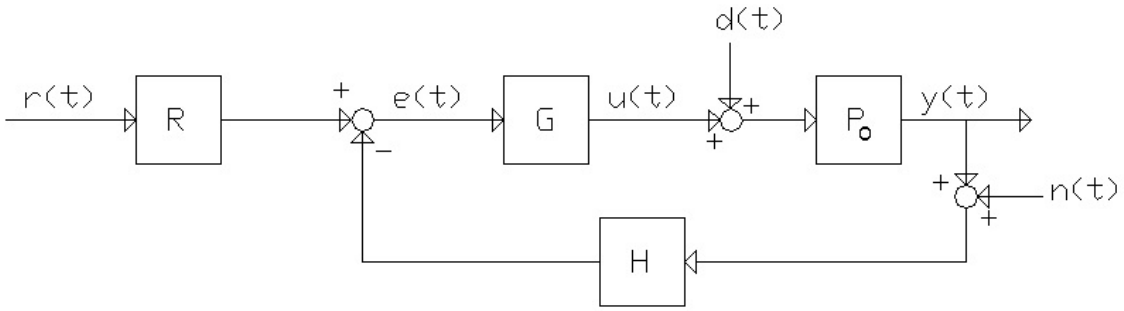


Figura 3.1: Schema di un sistema di controllo classico

La legge di controllo, cioè il legame tra e ed u, è:

$$u(t) = K_p e(t) + K_I \int_{t_0}^{t} e(\tau)d\tau + K_D \frac{de}{dt} \tag{3.1}$$

Applicando la trasformata di Laplace alla 3.1 con $t_0 = 0$ si ottiene la seguente funzione di trasferimento:

$$R_{PID}(s) = K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s} \tag{3.2}$$

Una diversa rappresentazione dei PID, forse anche più utilizzata della 3.2 è data da:

$$R_{PID}(s) = K_P(1 + \frac{1}{T_I s} + T_D s) = K_P \frac{T_I T_D s^2 + T_I s + 1}{T_I s} \tag{3.3}$$

in cui $T_I = \frac{K_P}{K_I}$ è il tempo integrale e $T_D = \frac{K_D}{K_P}$ è il tempo derivativo. Il PID ideale con funzione di trasferimento in 3.2 o 3.3 è un sistema con due zeri a parte reale negativa e un solo polo nell'origine. Il PID è quindi un sistema improprio per la presenza del termine derivativo $R_D(s) = K_D s$. Per questo motivo nella pratica l'azione derivativa è ottenuta per mezzo della funzione di trasferimento:

$$R_D^a(s) = \frac{K_p T_D s}{1 + \frac{T_D}{N} s} = \frac{K_D s}{1 + \frac{K_D}{K_P N} s} \tag{3.4}$$

dove la costante N è scelta in modo che il polo $s = -\frac{N}{T_D}$ aggiunto per la fisica realizzabilità sia all'esterno della banda di frequenze di interesse nel controllo. Valori tipici di N sono tra 5 e 20. Il PID in forma reale ha quindi la funzione di trasferimento:

$$R_{PID}^a(s) = K_P(1 + \frac{1}{T_I s} + \frac{T_D}{1 + \frac{T_D}{N} s} s) = K_P + \frac{K_I}{s} + \frac{K_D s}{1 + \frac{K_D}{K_P N} s} \tag{3.5}$$

Si osservi che il polo $s = -\frac{N}{T_D}$ di $R_D^a(s)$ modifica anche la posizione degli zeri della funzione di trasferimento complessiva del PID. Tuttavia, si può verificare che per N sufficientemente grande che gli zeri della 3.5 approsimativamente coincidono con quelli della 3.3.

Azione Proporzionale

L'azione proporzionale è ottenuta moltiplicando il segnale d'errore e con un'opportuna costante:

$$u_I = K_P e$$

È perfettamente possibile regolare un processo con un simile controllore, che risulta anche in grado di stabilizzare processi instabili. Tuttavia, non è possibile garantire che il segnale d'errore e converga a zero: questo perchè un'azione di controllo u è possibile solo se l'errore e è diverso da zero.

Azione Integrale

L'azione integrale è proporzionale all'integrale nel tempo del segnale di errore e, moltiplicato per la costante K_I:

$$u_I = K_I \int e(t)\, dt$$

Questa definizione dell'azione integrale fa sì che il controllore abbia memoria dei valori passati del segnale d'errore; in particolare, il valore dell'azione integrale non è necessariamente nullo se è nullo il segnale d'errore. Questa proprietà dà al PID la capacità di portare il processo esattamente al punto di riferimento richiesto, dove la sola azione proporzionale risulterebbe

nulla. L'azione integrale è anche l'elemento metastabile di un PID, perché un ingresso costante non convergerà a un determinato valore. Il fenomeno del *windup* è dovuto alla presenza dell'integratore.

Azione Derivativa

Per migliorare le prestazioni del controllore si può aggiungere l'azione derivativa:

$$u_D = K_D \frac{de}{dt}$$

L'idea è compensare rapidamente le variazioni del segnale di errore: se vediamo che l'errore e sta aumentando, l'azione derivativa cerca di compensare questa deviazione in ragione della sua velocità di cambiamento, senza aspettare che l'errore diventi significativo (azione proporzionale) o che persista per un certo tempo (azione integrale). L'azione derivativa è spesso tralasciata nelle implementazioni dei PID perché li rende troppo sensibili: un PID con azione derivativa, per esempio, subirebbe una brusca variazione nel momento in cui il riferimento venisse cambiato quasi istantaneamente da un valore a un altro, risultando in una derivata di e tendente a infinito, o comunque molto elevata. Ciò sconsiglia l'applicazione dell'azione derivativa in tutti i casi in cui l'attuatore fisico non deve essere sottoposto a sforzi eccessivi.

3.2 Pole placement

Il problema del controllo consiste nel progettare un controllore C che, sulla base delle informazioni fornite da un segnale di riferimento *r(t)* e uno di uscita *y(t)*, sia in grado di generare un segnale di controllo *u(t)* che faccia evolvere il sistema S in modo da minimizzare lo scostamento tra y e r. Lo schema classico di un sistema di controllo in retroazione è il seguente:

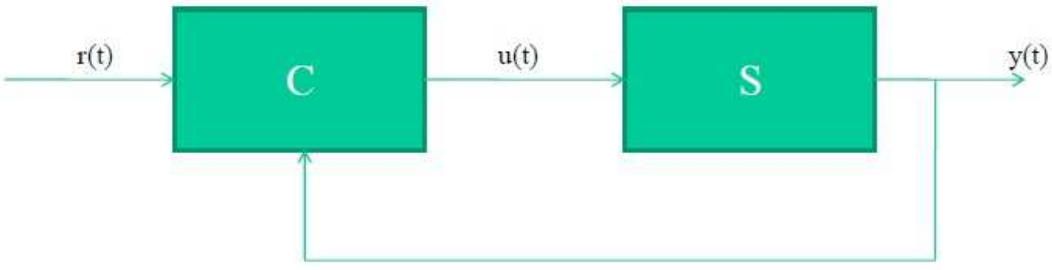


Figura 3.2: Schema di sistema di controllo in retroazione di uscita

Da quanto detto, è ovvio che l'informazione basata sull'uscita *y(t)* è incompleta, in quanto basata solo su una combinazione lineare delle variabili di stato. Un'informazione completa (*full information*), dovrebbe essere invece basata sulla conoscenza di tutte le variabili di stato. Per questo motivo, una generalizzazione della tecnica della controreazione può essere ottenuta utilizzando uno schema del genere:

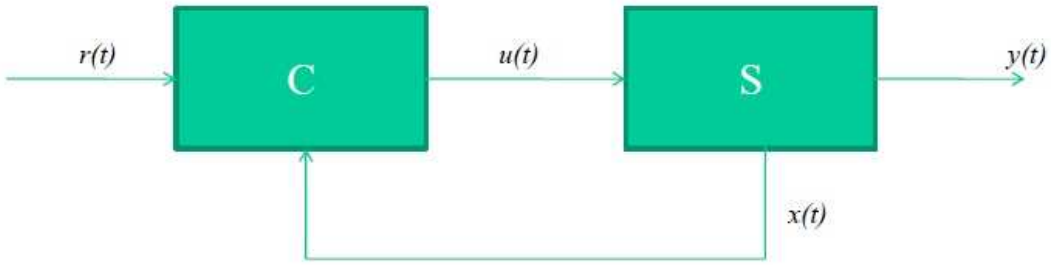


Figura 3.3: Schema di sistema di controllo in retroazione di stato

Questo schema ha trovato una larga utilizzazione nella risoluzione di problemi di controllo (stabilizzazione, allocazione dei poli, controllo ottimo, etc.). Ovviamente lo schema presentato è implementabile se tutto lo stato è misurabile.
Negli approcci summenzionati, oggetto di analisi durante il corso di studi, la struttura del controllore è quasi sempre molto semplice, riducendosi ad una combinazione lineare delle variabili di stato. Una prima tecnica di controllo che verrà analizzata è quella che si preoccupa semplicemente di progettare un controllore che stabilizzi il sistema a ciclo chiuso.
Per la scrittura formale dell'ingresso di controllo, definiamo prima un sistema LTI di ordine n con m ingressi come il seguente:

$$\dot{x} = Ax + Bu \qquad A \in \mathbb{R}^{nxn} \quad B \in \mathbb{R}^{nxm} \tag{3.6}$$

quindi il problema della stabilizzazione con retroazione lineare dello stato si può porre come segue.

Stabilizzazione

In riferimento allo schema presentato in Figura 3.3, il controllore C, finora visto come una *black box*, risponde alla necessità di trovare una legge di controllo del tipo:

$$u = Kx + r\ , \qquad K \in \mathbb{R}^{mxn} \tag{3.7}$$

tale che il sistema a ciclo chiuso sia asintoticamente stabile. Con riferimento alla Figura 3.4, si vuole trovare una matrice K ammissibile tale che il sistema a ciclo chiuso nella forma:

$$\dot{x} = (A + BK)x + Br \tag{3.8}$$

sia asintoticamente stabile, ovvero tale che gli autovalori della matrice a ciclo chiuso $(A + BK)$ siano tutti a parte reale negativa. In tal caso, il sistema si dice essere *stabilizzabile*.

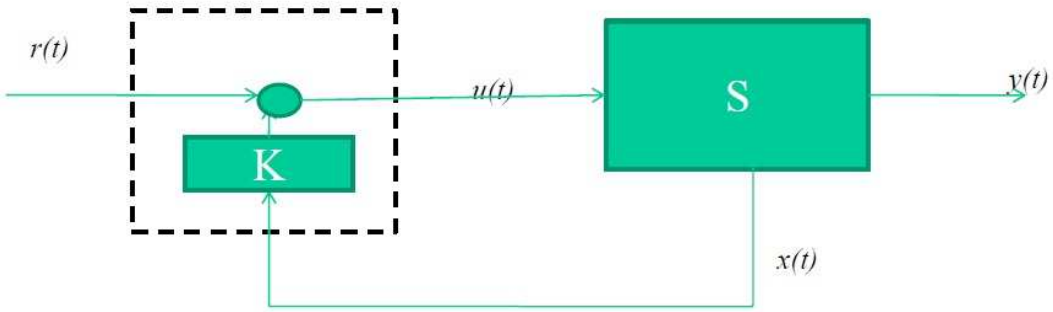


Figura 3.4: Schema di sistema di controllo in retroazione di stato con regolatore in dettaglio

Inoltre, si noti che la linearità e staticità del controllore rende molto semplice la sua implementazione.

Prestazioni

Risulta evidente che la semplice stabilizzazione del sistema a ciclo chiuso, sebbene fondamentale per il controllo dell'intero processo, non è l'unico obiettivo che si prefigge un controllore con retroazione di stato. Infatti, tipicamente un impianto fisico da controllare è già strutturalmente asintoticamente stabile, quindi lo scopo dell'azione di controllo risiede più che altro nel miglioramento delle sue prestazioni. In particolare, il sistema di controllo a ciclo chiuso, oltre alla asintotica stabilità, deve garantire il soddisfacimento di alcuni requisiti riguardanti le prestazioni, in termini di precisione a regime e comportamento in transitorio. Quest'ultimo può essere modificato se si riescono ad assegnare poli e zeri del sistema a ciclo chiuso.
Sia il soddisfacimento di tale comportamento in transitorio che la stessa stabilizzazione dell'intero sistema possono essere ottenuti attraverso l'utilizzo di opportune *LMI* (*Linear Matrix Inequalities*), ovvero Diseguaglianze Matriciali Lineari risolte per via numerica mediante algoritmi molto efficienti, grazie a strumenti di calcolo come il Matlab e che approfondiremo nei prossimi paragrafi.

3.3 LQR

La tecnica di controllo analizzata nel precedente paragrafo offre lo spunto di riflessione che porta subito a porsi l'interrogativo su quale sia il criterio di scelta per la collocazione dei poli ad anello chiuso del sistema, ovvero quale sia la migliore collocazione dei suddetti poli. In particolare, ci si riferisce alla soluzione *ottima* che, supposto un problema di regolazione, oltre a far tendere lo stato *x(t)* del sistema verso l'origine, si pone come ulteriore obiettivo anche la minimizzazione dell'utilizzo dell'ingresso di controllo, in un'ottica di economizzazione dell'uso degli attuatori. Generalmente, questi due obiettivi sono contrastanti e la soluzione più conveniente viene dalla risoluzione di un problema di controllo ottimo, ovvero una tecnica di piazzamento dei poli in maniera ottima.
Immaginando di ragionare in tempo discreto, dato un sistema dinamico:

$$\dot{x}(k) = A(k)x(k) + B(k)u(k) \tag{3.9}$$

con condizione iniziale *x(0)* assegnata, si sta cercando la sequenza ottima di ingressi che porta lo stato verso l'origine, minimizzando un indice di costo:

$$J = x(T)^T Q_T x(T) + \sum_{k=0}^{T-1} x(k)^T Q x(k) + u(k)^T R u(k) \tag{3.10}$$

dove Q e Q_T sono matrici simmetriche semidefinite positive, R è una matrice simmetrica definita positiva.
Si dimostra che per ogni matrice Q semidefinita positiva e per ogni matrice R definita positiva esiste sempre una soluzione $u_{ott}(t)$ del problema di controllo ottimo LQR che minimizza l'indice di costo *J(x,u)*.
Il controllo LQR permette di ottenere un controllo in retroazione dello stato ottimo rispetto all'indice quadratico nello stato *x(k)* e nel controllo *u(k)*, definito nella 3.10. Il controllore sintetizzato dipende dalla soluzione di una opportuna equazione di Riccati. Infatti, il controllo ottimo ottenuto u_{ott} è funzione lineare dello stato e di alcune matrici tra cui *P(t)* soluzione della DRE (equazione differenziale di Riccati) se il controllo è a tempo finito, o P (costante) soluzione della ARE (equazione algebrica di Riccati) se il controllo è a tempo infinito.

Tempo finito

Il controllo ottimo a tempo finito è il seguente:

$$u_{ott}(k) = -K_{ott}(k)x(k)$$

dove:

$$K_{ott}(k) = R^{-1}B^T P(k)$$

L'equazione differenziale di Riccati che fornisce la P(t) è la seguente:

$$-\dot{P(k)} = A^T P(k) + P(k)A + Q - P(k)BR^{-1}B^T P(k)$$

Tempo infinito

Il controllo ottimo è il seguente:

$$u_{ott}(k) = -K_{ott}x(k)$$

dove:

$$K_{ott} = R^{-1}B^T P$$

In questo caso, a differenza del problema del controllo ottimo in tempo finito, la soluzione viene dalla equazione algebrica di Riccati seguente:

$$A^T P + PA + Q - PBR^{-1}B^T P = 0$$

La differenza tra il controllo a tempo finito e il controllo a tempo infinito sta nel far tendere all'infinito ($T \to \infty$) l'estremo superiore della sommatoria (nel tempo discreto) o dell'integrale (nel tempo continuo). L'effetto di un controllo su tempo infinito è un controllore stazionario (indipendente dal tempo), ovvero una matrice K_{ott}, costante e ottima rispetto all'indice che si

voleva minimizzare.
Infine, è possibile dimostrare che il controllo LQR è robusto di per sé per una gamma di variazioni parametriche.

3.4 Filtro di Kalman

Nella teoria del controllo, l'osservatore di stato è un sistema dinamico con lo scopo di stimare l'evoluzione di stato di un sistema da osservare. La conoscenza dello stato è necessaria per risolvere molti problemi legati al controllo; ad esempio per implementare leggi di controllo con feedback quando non si può misurare direttamente lo stato del sistema o anche quando, pur potendo misurarlo, l'errore nella misurazione è più grande di quello che si commette andando a stimare lo stato del sistema. Affinché la stima sia possibile, occorre che il sistema di cui si vuole stimare lo stato sia osservabile, e quindi, che la matrice test di osservabilità abbia rango pari all'ordine del sistema. In tal modo, è possibile ricostruire lo stato del sistema a partire dall'osservazione dell'output.
A questo scopo, per la caratterizzazione dello stato di un sistema dinamico affetto da rumore e a partire da una serie di misure soggette a rumore, si utilizza il filtro di Kalman, un efficiente stimatore (l'analogo stocastico dell'osservatore di Luenberger in ambito deterministico), che sotto determinate ipotesi risulta essere uno stimatore ottimo.
Dato un sistema dinamico lineare tempo invariante soggetto a rumore di processo $v_x(t)$, e rumore di misura $v_y(t)$, si scrivono le equazioni caratteristiche come:

$$\begin{aligned} \dot{x} &= Ax(t) + Bu(t) + v_x(t) \\ y &= Cx(t) + v_y(t) \end{aligned}$$

dove i rumori $v_x(t)$ e $v_y(t)$ sono incorrelati nel tempo, gaussiani e a media nulla. Dato il rumore, si scrive una matrice V di covarianza:

$$V = \begin{pmatrix} Q & Z \\ Z & R \end{pmatrix} \tag{3.11}$$

dove $Z = 0$, ovvero i rumori su stato e uscita sono incorrelati, lo stato è modellizzato come una variabile casuale gaussiana tale che:

$$\begin{aligned} &x_0 = x(0) \\ &E[x_0] = \bar{x}_0 \\ &E[(x_0 - \bar{x}_0)(x_0 - \bar{x}_0)^T] = \tilde{P}_0 \geq 0 \end{aligned}$$

e inoltre i rumori e lo stato sono incorrelati. A questo punto si considera l'osservatore:

$$\frac{d\hat{x}(t)}{dt} = \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + L(t)[y(t) - C\hat{x}(t)] \tag{3.12}$$

da cui, con alcuni passaggi algebrici, si ricava la dinamica dell'errore:

$$\begin{aligned} \dot{\hat{e}}(t) &= \dot{x}(t) - \dot{\hat{x}}(t) \\ \dot{\hat{e}}(t) &= A_c(t)e(t) + B_c(t)v(t) \end{aligned}$$

dove:

$$A_c(t) = A - L(t)C$$
$$B_c(t) = I - L(t)$$

Scrivendo:

$$\bar{e}(t) = E[e(t)]$$
$$\dot{\bar{e}}(t) = A_c(t)\bar{e}(t) + B_c(t)E[v(t)] = A_c(t)\bar{e}(t)$$

in cui:

$$v(t) = \begin{pmatrix} v_x(t) \\ v_y(t) \end{pmatrix} \tag{3.13}$$

si nota che il valor atteso dell'errore è un sistema autonomo. Definita la covarianza dell'errore $\tilde{P}(t)$, l'obiettivo di ottimizzazione è quello di trovare $L(t)$ che minimizza la cifra di merito:

$$min_{L(t)}\gamma^T\tilde{P}(t)\gamma \tag{3.14}$$

Si dimostra che la $L(t)$ che risolve il problema di ottimizzazione è:

$$L(t) = \tilde{P}(t)C^T\tilde{R}^-1 \tag{3.15}$$

dove $P(t)$ è soluzione dell'equazione di Riccati:

$$\dot{\tilde{P}}(t) = A\tilde{P}(t) + \tilde{P}(t)A^T + \tilde{Q} - \tilde{P}(t)C^T\tilde{R}^-1C\tilde{P}(t). \tag{3.16}$$

3.5 Applicazione delle strategie di controllo

Finora ci siamo soffermati sull'analisi generale delle strategie di controllo principali studiate durante il corso di studi di Sistemi di Controllo Multivariabile. Nei presenti paragrafi, andremo ad approfondire ciascuna di queste tecniche in relazione alle loro applicazioni sul sistema in esame, ovvero il modello di quadricottero. In particolare si esamineranno prima le singole tecniche di controllo, poi nel capitolo successivo si metteranno in luce aspetti positivi e negativi, con il confronto delle une con le altre, per giudicare quella ottimale, ossia quella che garantisce il miglior rispetto delle specifiche date.

3.5.1 Applicazione del PID

Come prima tecnica di controllo attuata si analizza quella più classica e diffusa nel campo dei controllo, ovvero il controllo PID. In riferimento al modello matematico del quadricottero analizzato nel capitolo 2, si è pensato di sfruttare una caratteristica del modello per fornire gli ingressi retroazionati del sistema, opportunamente moltiplicati per le rispettive costanti proporzionali (K_P) integrali (K_I) e derivative (K_D). In particolare, nel modello matematico si è visto che la differenza di velocità dei motori sullo stesso asse contribuiscono a creare coppia non nulla relativa all'asse ortogonale a quello su cui sono posizionati i motori stessi; infatti:

$$\tau_\phi = lk(-\omega_2^2 + \omega_4^2)$$
$$\tau_\phi = lk(-\omega_1^2 + \omega_3^2)$$

Quindi, abbiamo pensato di sfruttare questa caratteristica per controllare l'assetto del quadricottero. Ragionamento analogo è stato fatto per l'angolo di imbardata ψ, per il quale però viene portata in conto la somma di tutti e quattro i motori. Di seguito forniamo lo schema di realizzazione Simulink usato per le simulazioni.

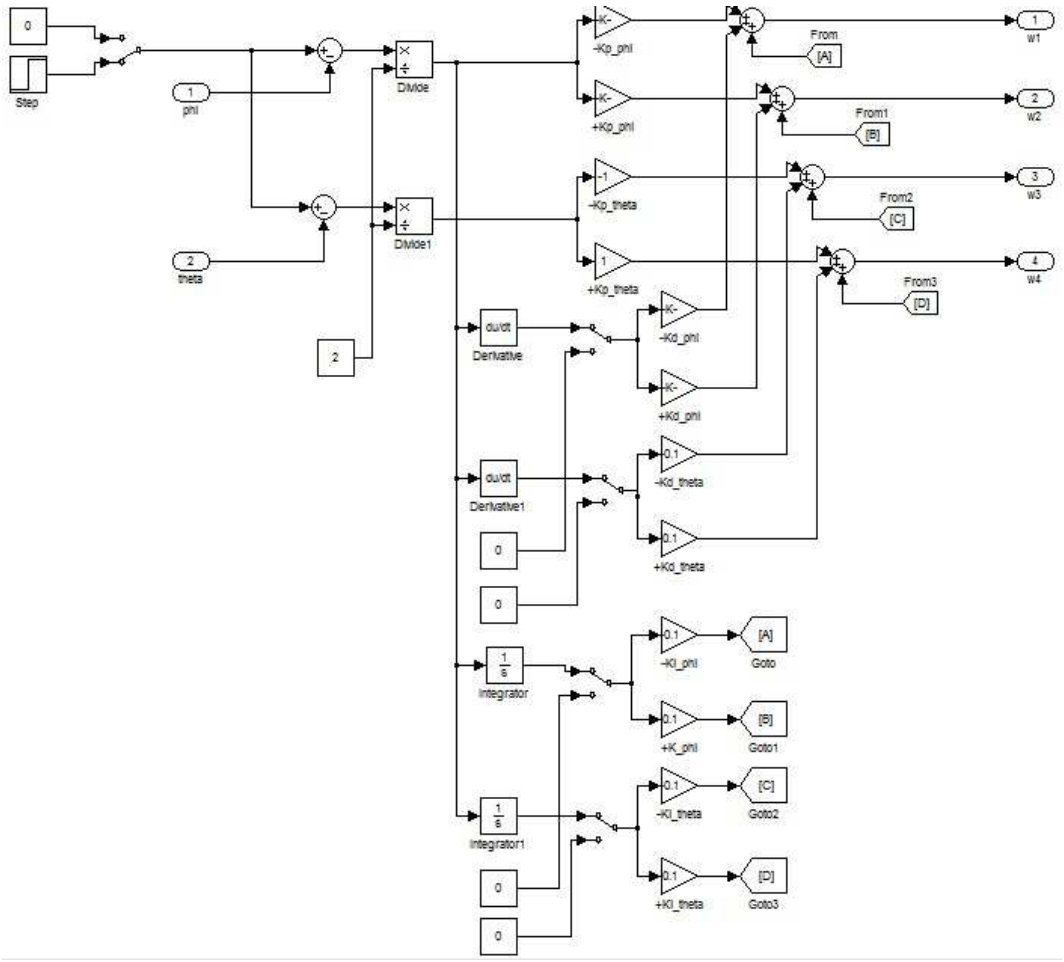


Figura 3.5: Schema di sistema di controllo PID

3.5.2 Applicazione del Pole Placement

In questa sezione si analizza il blocco di controllo che garantisce l'assegnamento dei poli del sistema a ciclo chiuso all'inteerno di sezioni del piano complesso, nel rispetto delle specifiche di progetto. In particolare, il Matlab mette a disposizione la funzione $K = PLACE(A, B, P)$ in cui K è la matrice di reetroazione di stato tale che gli i poli del sistema a ciclo chiuso $A - BK$ siano quelli specificati da P. Lo schema di controllo sviluppato in Simulink è identico a quello visto in questo capitolo in figura 3.3.

3.6 Applicazione delle strategie di controllo

Finora ci siamo soffermati sull'analisi generale delle strategie di controllo principali studiate durante il corso di studi di Sistemi di Controllo Multivariabile. Nel presente paragrafo, andremo ad approfondire ciascuna di queste tecniche in relazione alle loro applicazioni sul sistema in esame, ovvero il modello di quadricottero.

3.6.1 Strategia di controllo LQR con piena retroazione delle stato *FULL STATE FEEDBACK*

Lo scopo di costruire un controllore LQR immaginando di avere *accessibile* tutto lo stato è molto utile in modo da avere un valido ed ideale termine di paragone per una realistica strategia di controllo dove non tutto lo stato è misurabile e pertanto si deve ricorrere ad una stima di questo. Per effetture delle simulazioni abbiamo costruito due schemi implementativi un primo dove è stato costruito un unico controllore LQR con azioni in avanti che non solo stabilizza il sistema ma paermette anche inseguinmento. Abbiamo poi sviluppato un secondo controllo LQR ragionando solo sulle variabili del software di controllo dove ci sarebbe stato possibile andare implementare il controllore. Le variabili su cui abbiamo effettuato il controllo sono z, ϕ, θ, ψ.
Il primo controllore costruito è implementato nel seguente schema Simulink:

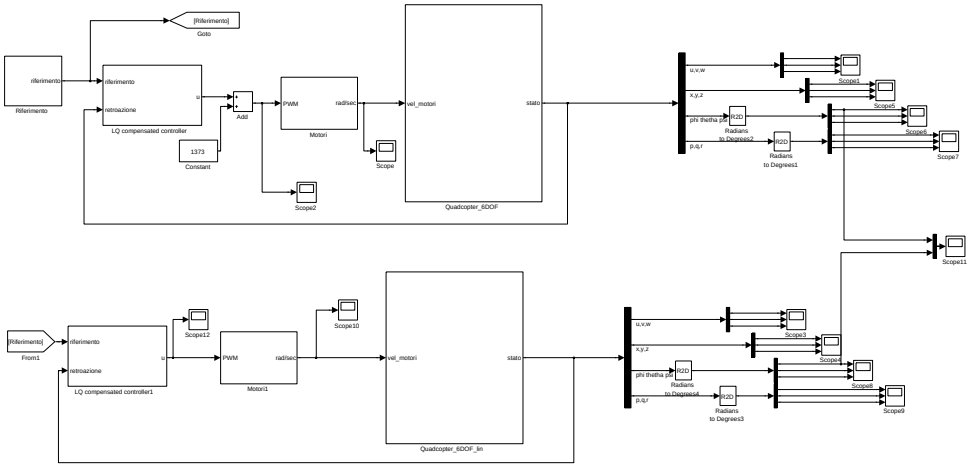


Figura 3.6: Schema di controllo Lqr

Si osservi che sono state modellate anche le dinamiche degli attuatori come mostrato nel capitolo 2 e che il controllore è stato applicato in questo caso al sistema non lineare sebbene in altre fasi sia lavorato ovviamente sul sistema lineare.Quanto detto verrà ovviamente mostrato tramite simulazioni nel capitolo 4. Nel blocco del controllore troviamo infatti sia l'azione in retroazione sia l'azione in avanti che permette l'inseguimento.

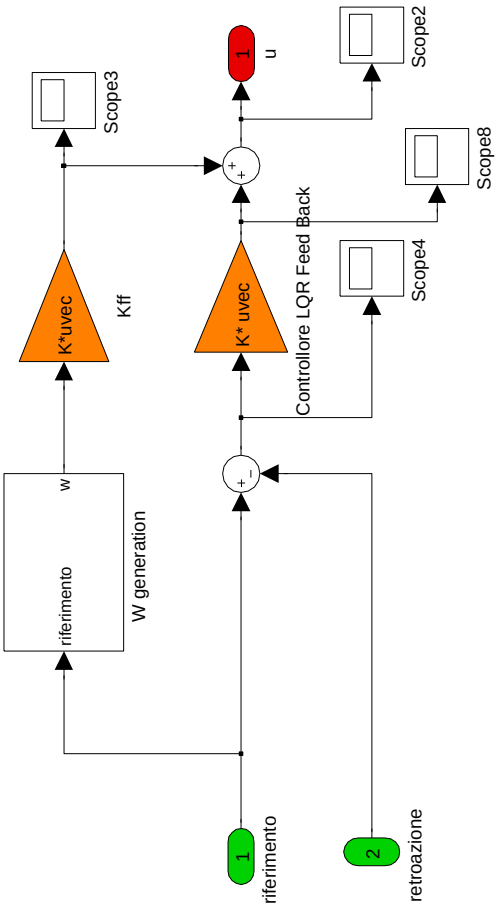


Figura 3.7: Controllore Lqr

In questo schema le matrici Q ed R sono state scelte usando la cosidetta *Bryson rule* che dice di definire gli elementi interessati nelle matrici come:

$$\frac{1}{x_{max}^2} \qquad \frac{1}{u_{max}^2} \tag{3.17}$$

dove x_{max} ed u_{max} sono la massima variazione che si è disposti a tollerare. Chiaramente il processo ha richiesto rifinitura e dopo un breve processo iterativo siamo arrivati a definire le matrici Q ed R come segue:

$$Q = diag(1, 1, 5, 10, 15, 20, 20, 20, 20, 15, 15, 15) \quad R = diag(0.1, 0.1, 0.1, 0.1) \tag{3.18}$$

Al fine di costruire il controllore abbiamo scritto una funzione che restituisce in output i guadagni in retroazione ed in avanti per il controllore Successivamente al controllore a 12 stati abbiamo costruito più controllori LQR sulle variabili di stato che il software di basso livello ci permetteva di costruire. I controllori usati sono allora stati sviluppati estraendo dalla matrice A le righe interessate e su di questi è stato risolto un apposto problema LQ con azione in avanti. La prima variabile che siamo andati a controllare è la variabile di altitudine z assieme alla variabile w che mostra la variazione di altitudine. Lo schema usato è il seguente (dove si osservi che non è inserito l'impianto lineare bensi il non lineare sebbene nelle simulazioni verrà fatto un confronto):

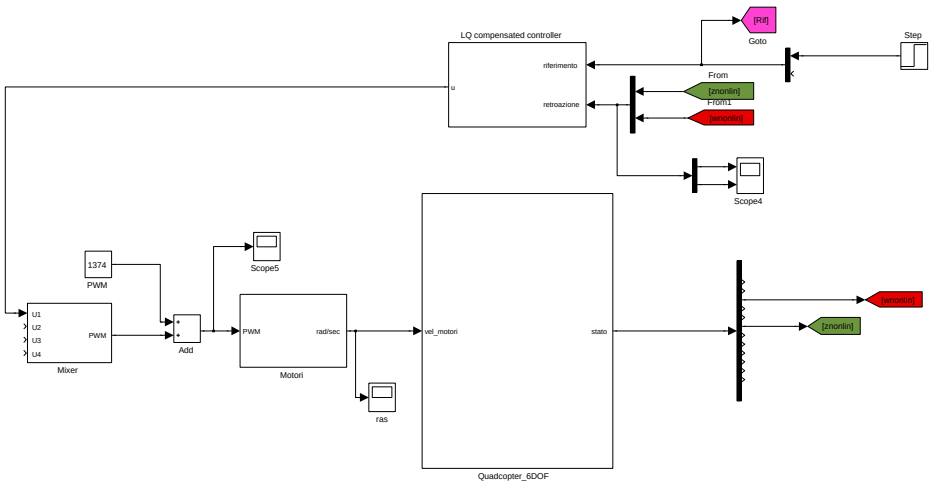


Figura 3.8: Controllore altitudine

In questo schema si è utilizzato il mixer introdotto al capitolo 2.
Oltre a z con questa logica di progetto sono stati costruiti anche i controllori per l'assetto ϕ,θ,ψ i cui diagrammi sono mostrati nei seguenti grafici:phicontrol

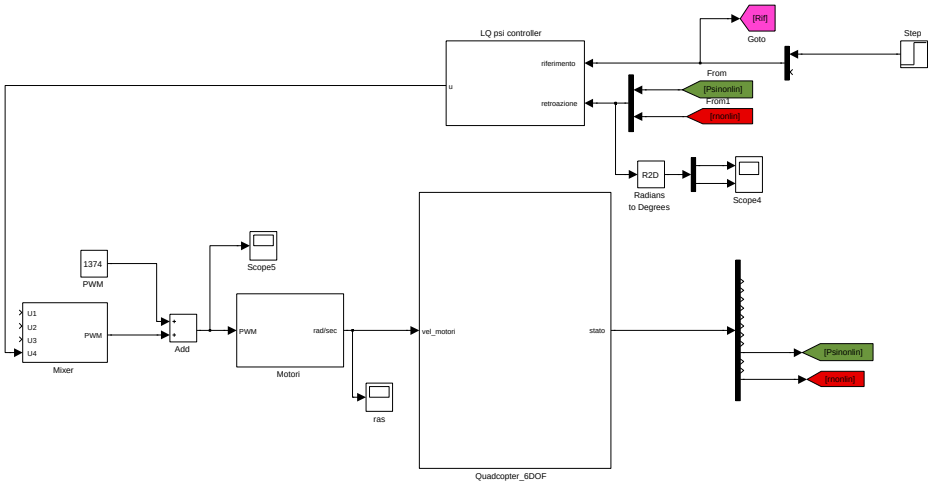


Figura 3.9: Controllore ϕ

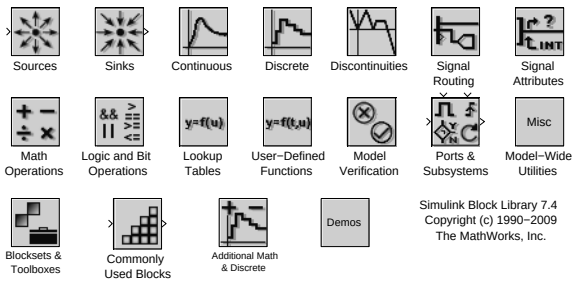


Figura 3.10: Controllore ϕ

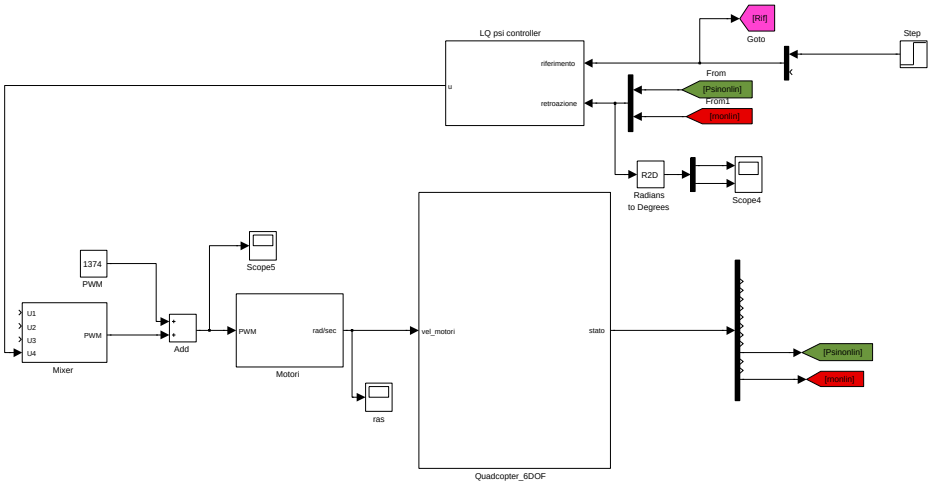


Figura 3.11: Controllore ψ

Il listato usato per costruire le matrici sulle quali abbiamo risolto il problema LQ è il seguente: Lo schema complessivo su cui sono stati provati i controllori è il seguente dove abbiamo valutato l'effetto dei vari controllori sul sistema complessivo.

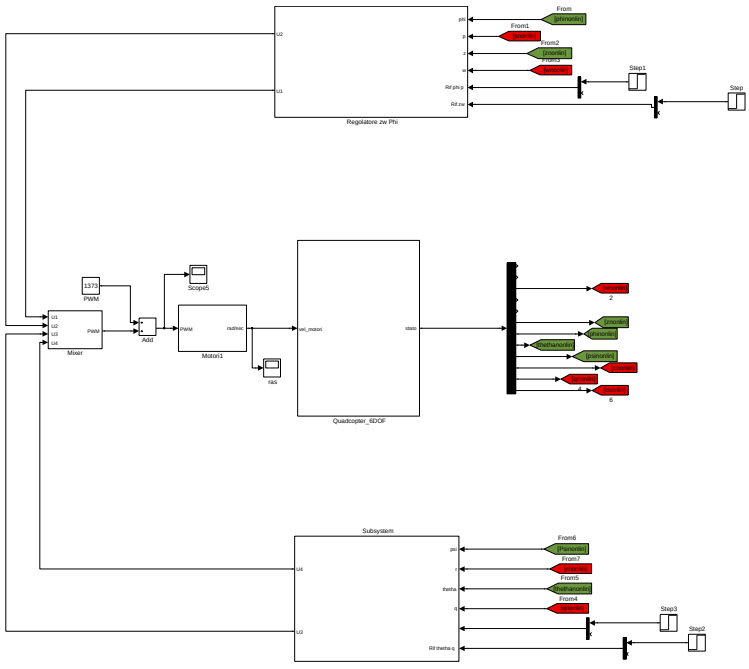


Figura 3.12: Schema complessivo

3.6.2 Strategia di controllo FSI con controllo LQ a TD

Per verificare il controllore abbiamo anche costruito uno schema a tempo discreto risolvendo il problema LQ a TD. I risultati sono stati soddisfacenti anche in questo caso ,dov, è ben noto che il campionatore e il mantenitore del sistema di controllo introducono un ritardo che, *deve* poter essere tollerato dal sistema. E' stata pertanto portata a termine anche un sintesi di controllori usando il problema LQ a TD. Lo schema rispetto al tempo continuo è invariato a meno del convertitore ADC che mostriamo qui di seguito.

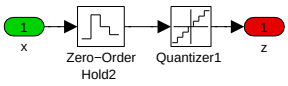


Figura 3.13: Convertitore *ADC*

Lo schema utilizzato per la simulazione è lo stesso di fig.3.12 con la sola aggiunta del campionatore in uscita al sistema. E' chiaro anche qui che il confronto con il sistema lineare è stato fatto sebbene lo schema non sia riportato.

Capitolo 4

Simulazioni

Simulazioni

4.1 *Open loop*

Vediamo il grafico del sistema senza controllo, in open loop prima per gli angoli e poi per le posizioni.

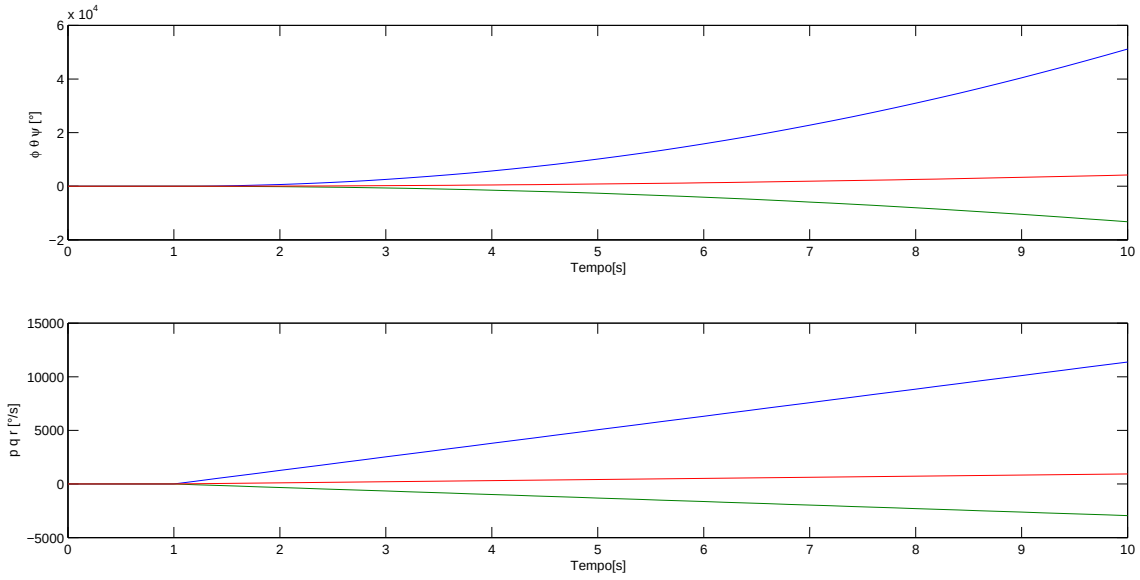


Figura 4.1: Posizioni angolari e velocitÃ angolari

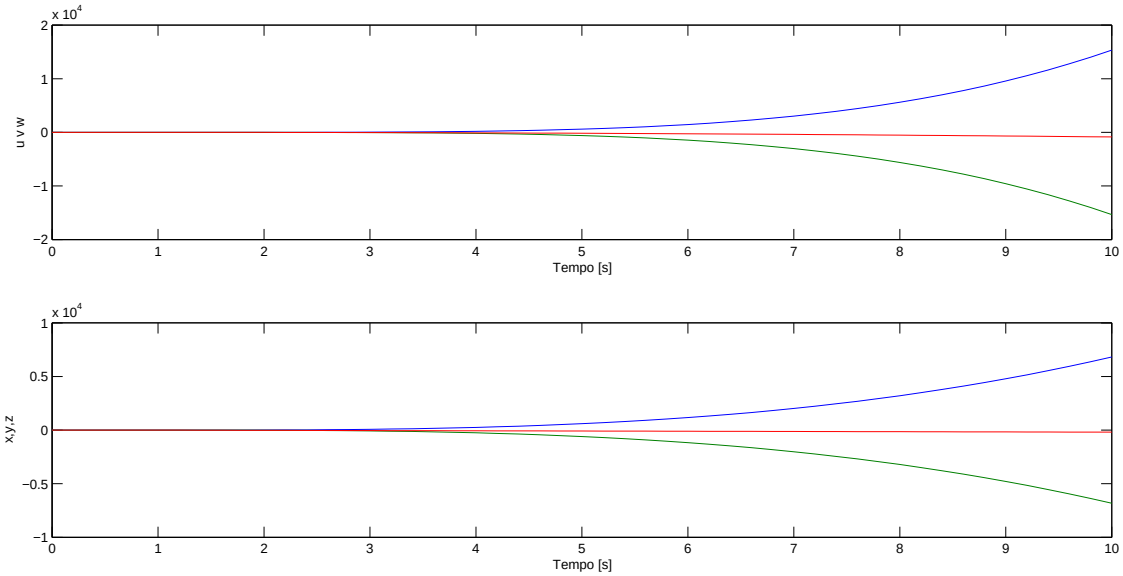


Figura 4.2: Posizioni e velocitÃ lineari

4.2 Simulazioni PID

Come prima simulazione del sistema controlloat mostriamo la risposta del sistema a un valore di riferimento dato da uno step che dal valore nullo passa ad un valore di 0.5 radianti. del puro controllo Proporzionale (K_P), in assenza quindi dei guadagni integrali (K_I) e derivativi (K_D):

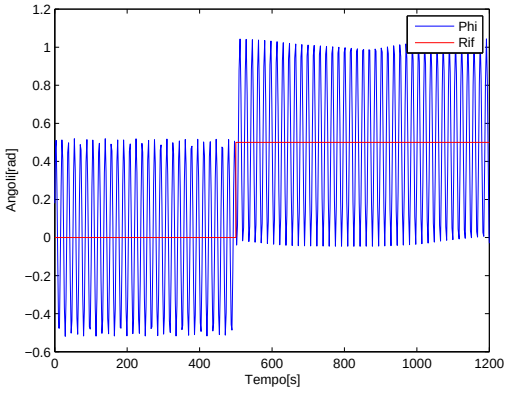


Figura 4.3: Angolo di rollio ϕ con il solo guadagno proporzionale K_P

Risulta evidente come la sola azione proporzionale non sia sufficiente a far raggiungere un valore costante all'angolo di rollio del quadricottero, che rimane in continua oscillazione armonica. A questo punto, il passo successivo Ãš l'inserimento degli altri due guadagni K_I e K_D, con il seguente risultato:

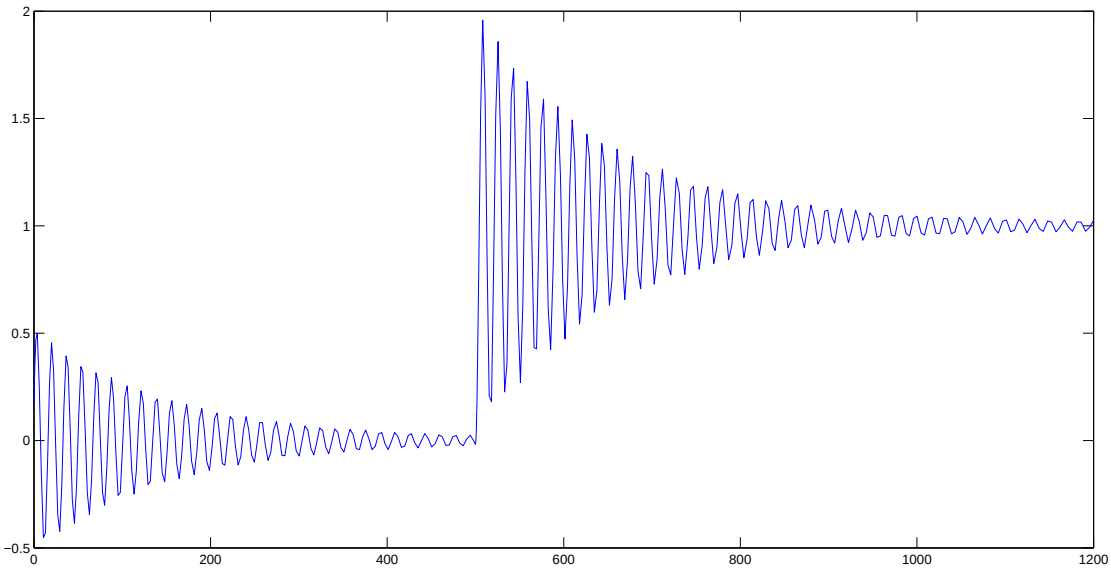


Figura 4.4: Angolo di rollio ϕ

in cui si nota che, pur con un tempo di assestamento elevato e con oscillazioni elevate, il velivolo va verso il valore di regime costante del riferimento. Si omettono gli andamenti temporali delle dinamiche di beccheggio e imbardata e livello in quanto qualitativamente equivalenti alla dinamica di rollio appena vista.

4.3 Pole Placement

La prima simulazione Ãš quella del controllo dell'angolo di rollio con un ingresso di riferimento di circa un decimo di radiante, posto a 3 secondi. Il piazzamento dei poli è stato fatto in modo da garantire che gli autovalori della matrice a ciclo chiuso $A - BK$ fossero tutti a sinistra dell'asse immaginario del piano complesso, con $Res \leq \alpha$, con alfa tale da soddisfare opportuni requisiti di rapiditÃ di convergenza (si è posto $\alpha = 6$). La prima simulazione Ãš quella del controllo dell'angolo di rollio con un ingresso di riferimento di circa un decimo di radiante, posto a 3 secondi, con i poli imposti tramite la funzione *place()* del Matlab a -0.5 e -1.5 (in riferimento al modello di controllo a blocchi discusso nel capitolo precedente):

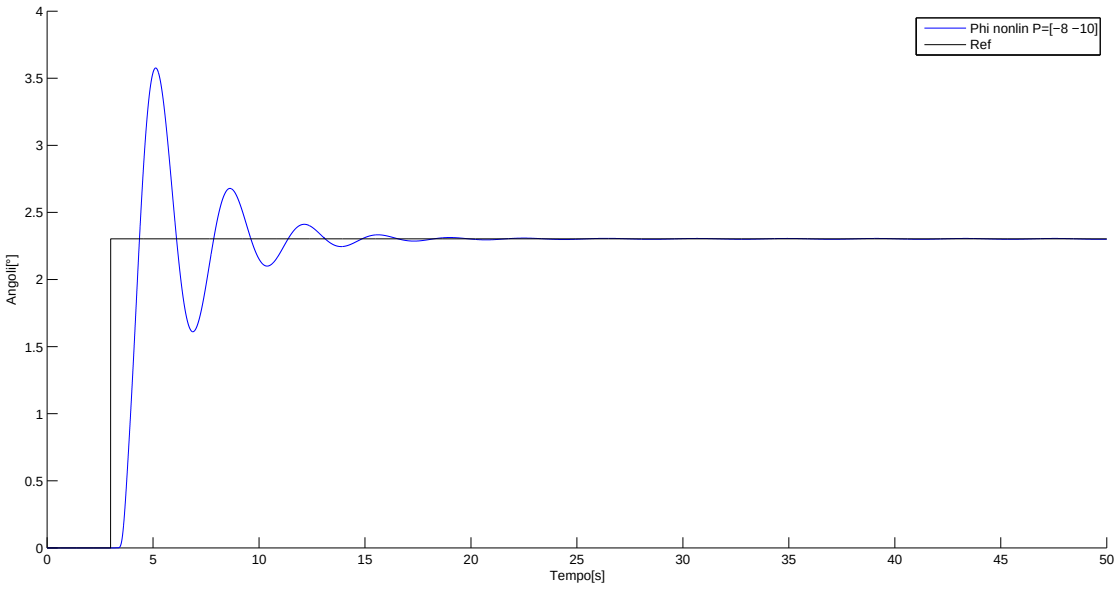


Figura 4.5: Rollio ϕ

Come si nota dal grafico, l'oscillazione iniziale è dovuta all'aver forzato il sistema ad inseguire il riferimento. Nelle prossime simulazioni invece, si mette in evidenza come si possa attenuare questo fenomeno, andando a posizionare i poli del sistema a ciclo chiuso un po' più vicino all'origine. Infatti, per l'angolo di beccheggio si è ottenuta la seguente simulazione:

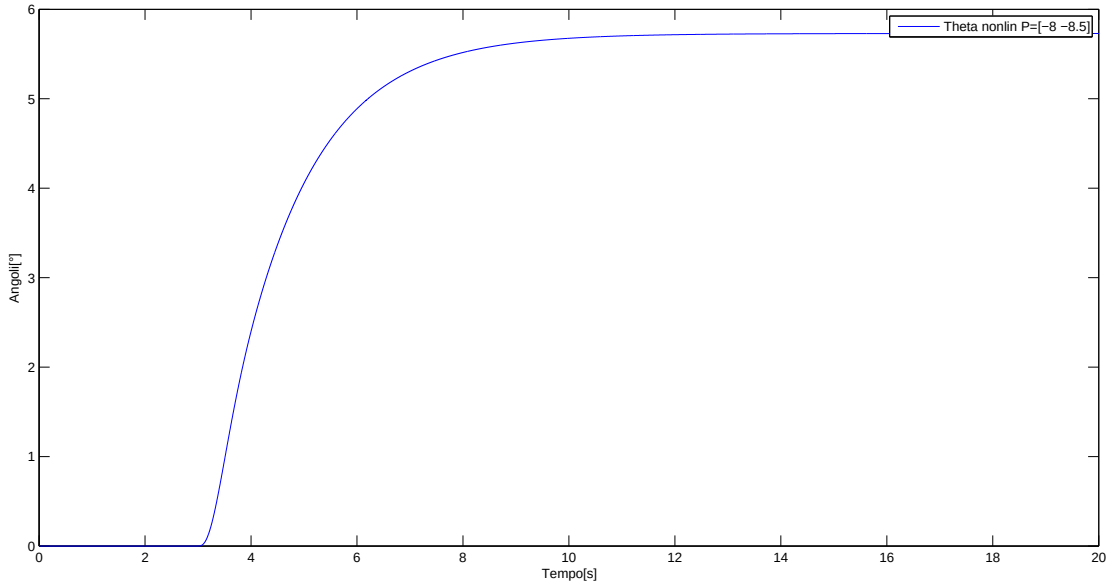


Figura 4.6: Beccheggio θ

con poli piazzati a -8 e 8.5 e per l'angolo di imbardata si è imposto un riferimento a step pari a 1 radiante con il seguente risultato:

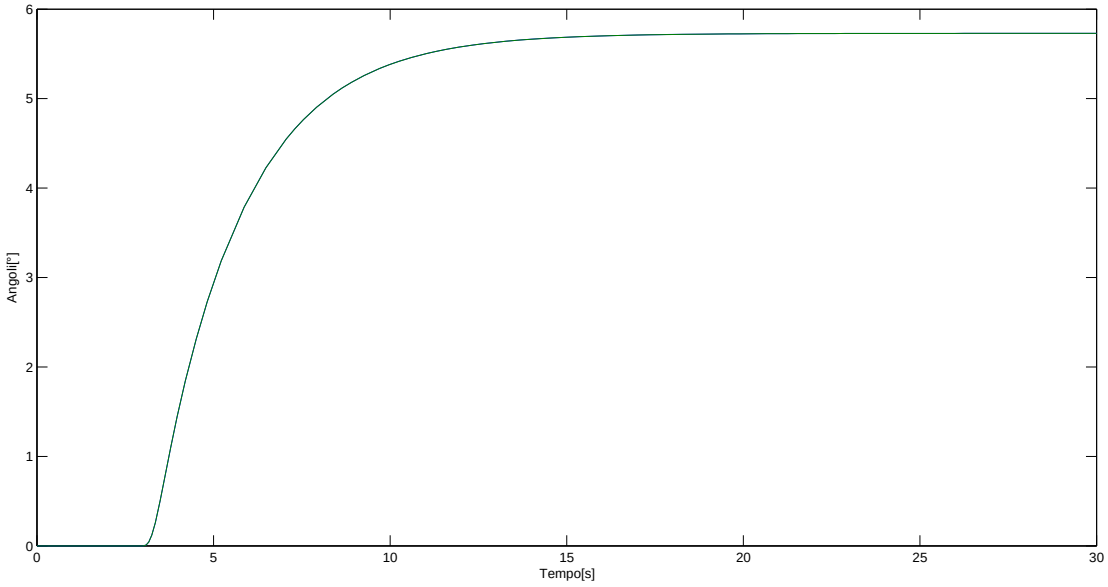


Figura 4.7: Imbardata ψ

con gli stessi poli usati per il *place* dell'imbardata. Inoltre, si noti che nel grafico siano presenti gli andamenti temporali del caso linearizzato e non lineare, pur risultando quasi completamente sovrapposti. Infine, il controllo di altitudine ha fornito il seguente risultato:

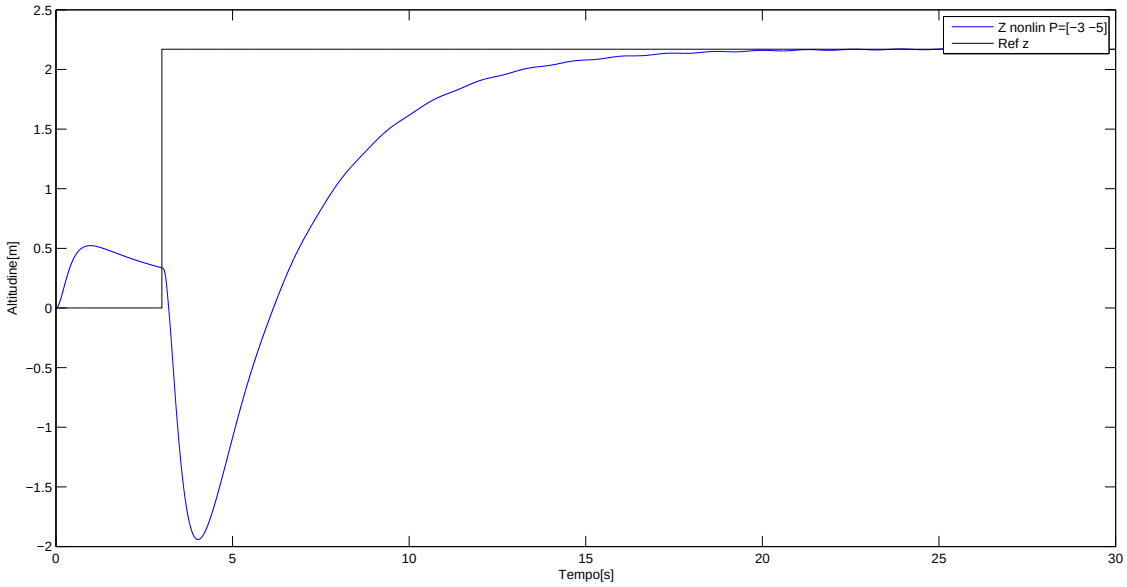


Figura 4.8: Altitudine z

con un riferimento a gradino di 2 metri, con piazzamento dei poli in -3 e -5. Nei paragrafi successivi si vedrà come la soluzione del problema LQR fornisca risultati migliori in termini di rapidità di convergenza.

4.4 Simulazioni LQ a TC FULL STATE INFROMATION

Step e confronto altitudine:

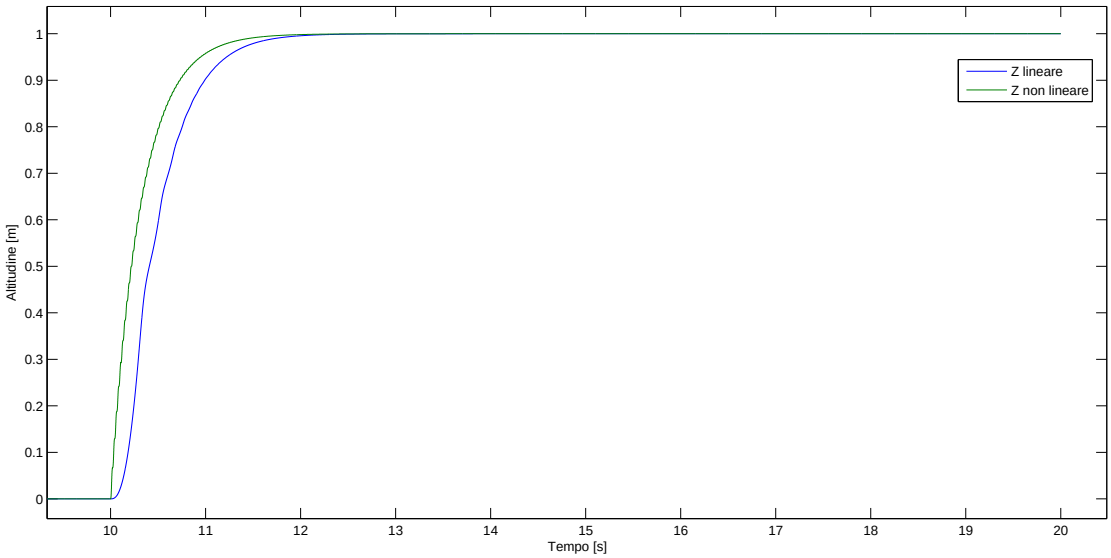


Figura 4.9: Altitudine z

Step di circa 5Â° tra sistema lineare e non per l'angolo di rollio.

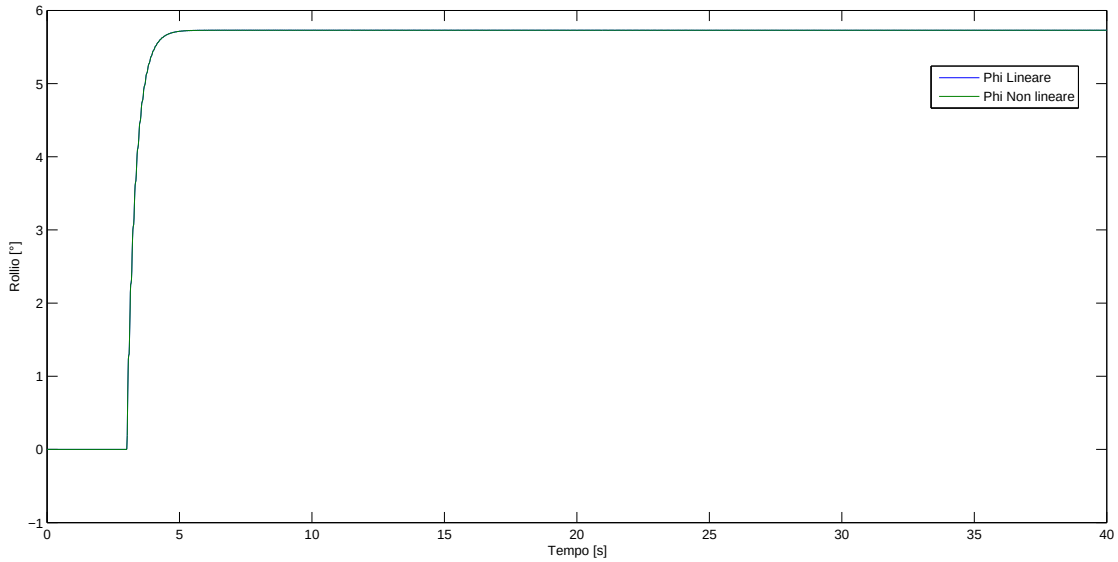


Figura 4.10: Angolo di rollio ϕ

Dinamiche degli attuatori per il controllore a tempo continuo

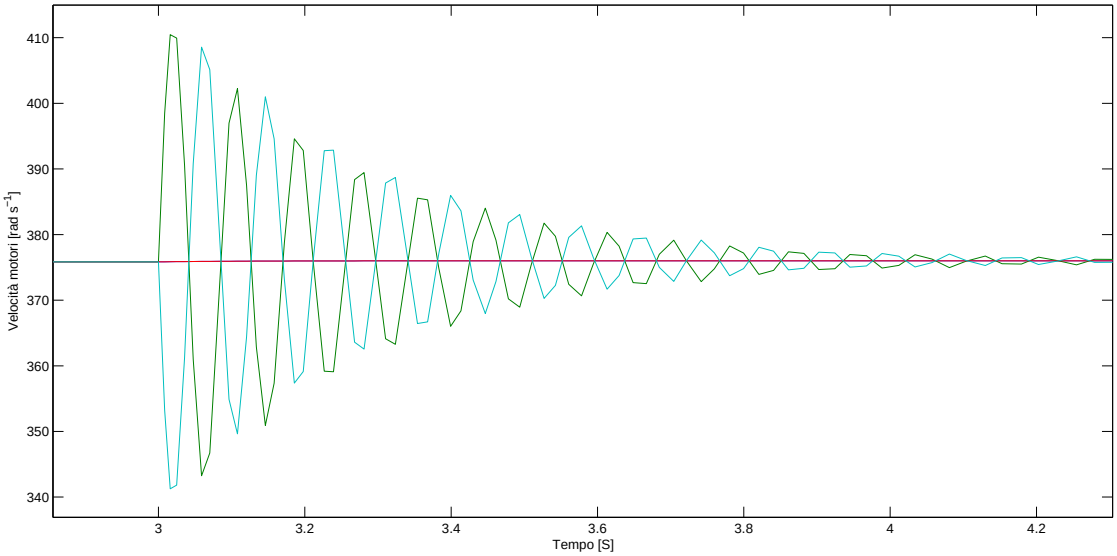


Figura 4.11: Diagramma attuatori per risposta a scalino su ϕ

Step di circa 5Â° tra sistema lineare e non per l'angolo di beccheggio.

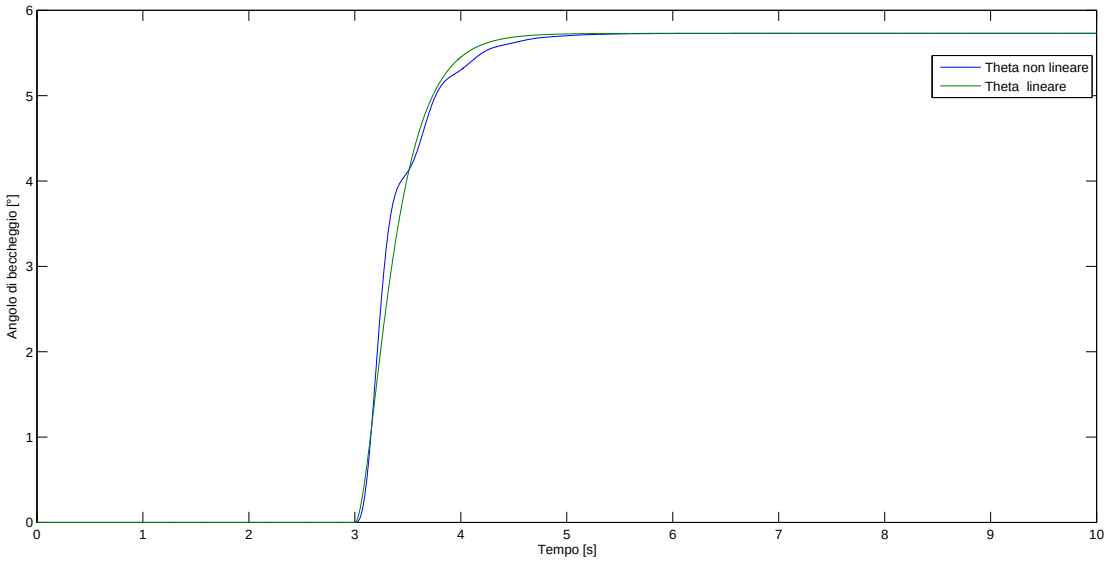


Figura 4.12: Angolo di rollio θ

Passiamo infine a vedere uno step di circa 0.2 radianti all'angolo di imbardata.

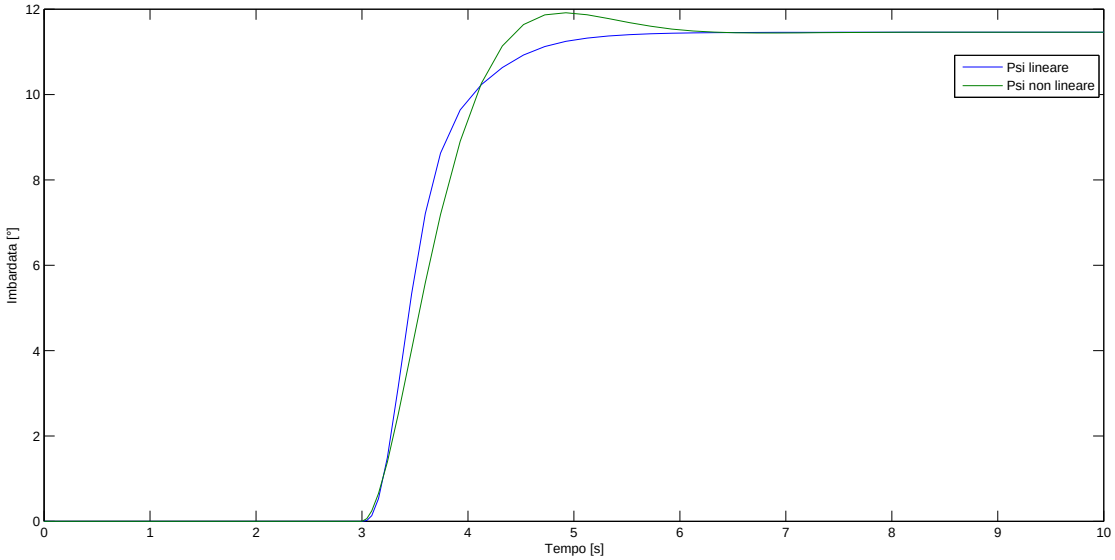


Figura 4.13: Angolo di rollio ψ

Per concludere vediamo un esempio di manovra

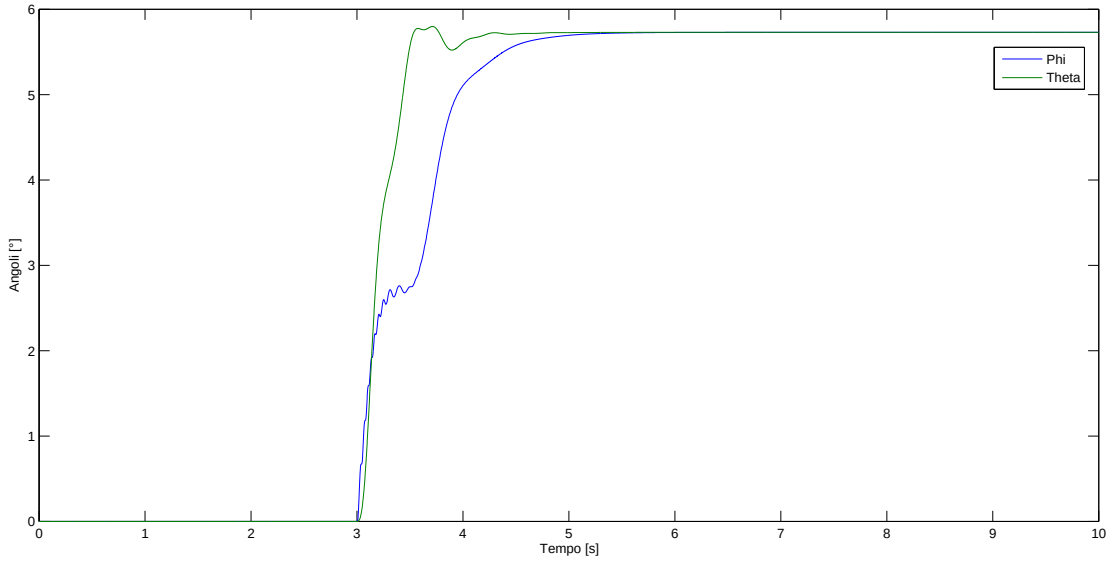


Figura 4.14: Manovra con step di 0.1 radianti ingresso su phi e ψ

4.5 Simulazioni LQ a TD FULL STATE INFROMATION

Vediamo alcune simulazioni per il controllo LQ a TD. La prima mostra uno step di circa 5Â° sull'angolo di rollio e le successive due mostrano simulazioni per gli angoli di beccheggio e imbardata.

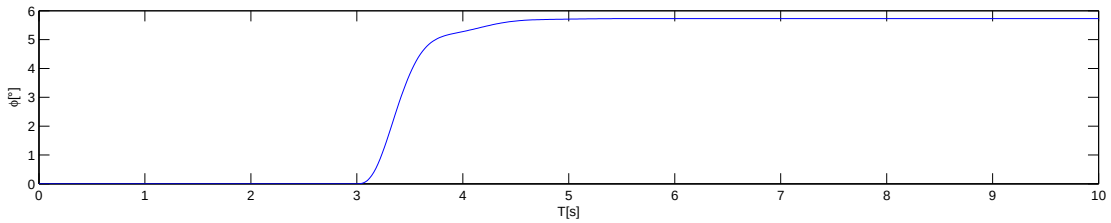


Figura 4.15: Step di 0.1 radianti ingresso su phi e ϕ

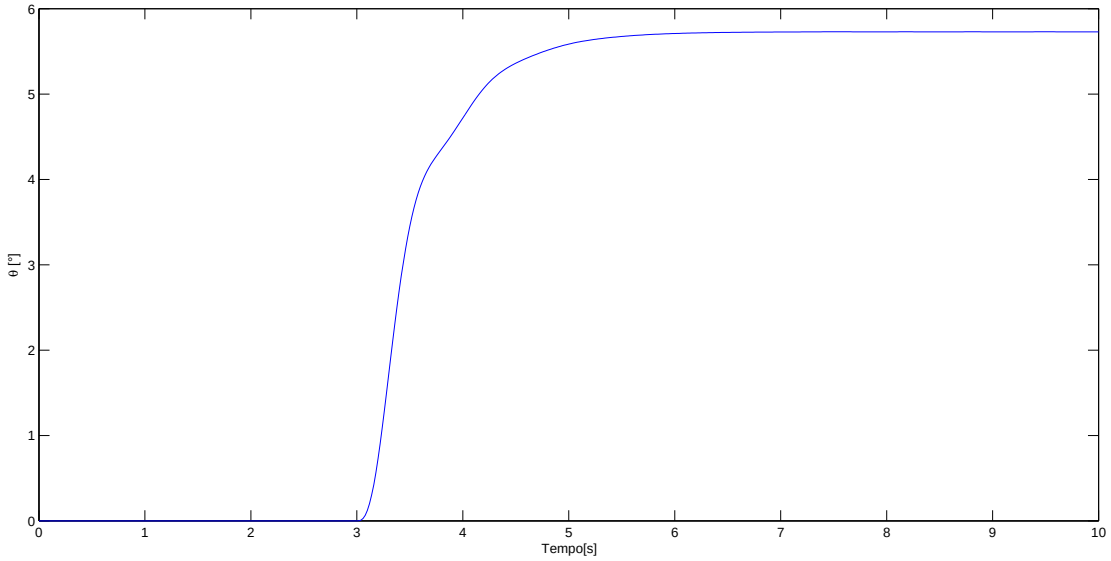


Figura 4.16: Step di 0.1 radianti ingresso su ϕ e θ

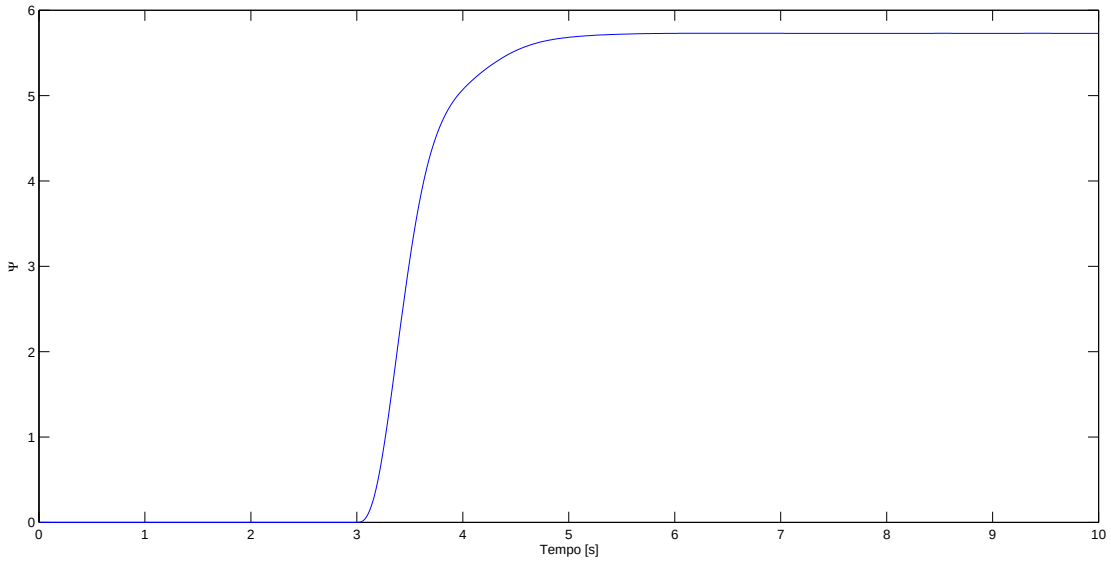


Figura 4.17: Step di 0.1 radianti ingresso su ϕ e ψ

Infine mostriamo un manovra sempre con step da 0.1 radianti:

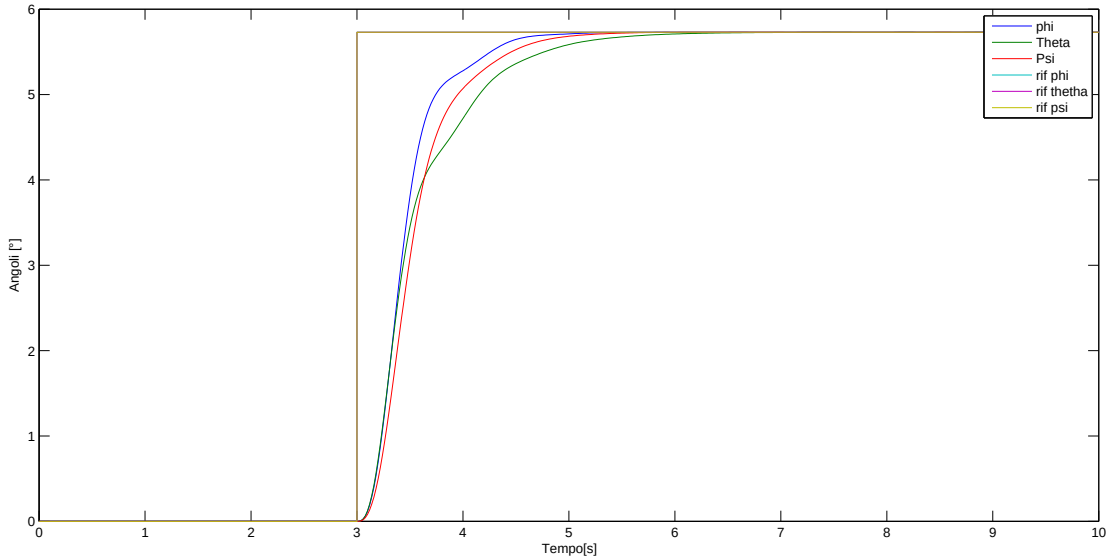


Figura 4.18: Manovra controllo TD su ϕ, θ, ψ

4.6 Simulazioni LQ con Filtro di Kalman

Questo paragrafo è dedicato alle simulazioni con controllo LQ e Filtro di Kalman per la stima dello stato. Si parte dicendo che sullo stato e sull'uscita sono presenti rumori stocastici bianchi gaussiani, rispettivamente di processo e di misura, con le matrici di covarianza assegnate tutte pari a un millesimo. La prima simulazione fatta riguarda il controllo dell'angolo di rollio con stato stimato, in presenza di uno step di riferimento di 0.1 radianti posto a 3 secondi, con il seguente risultato:

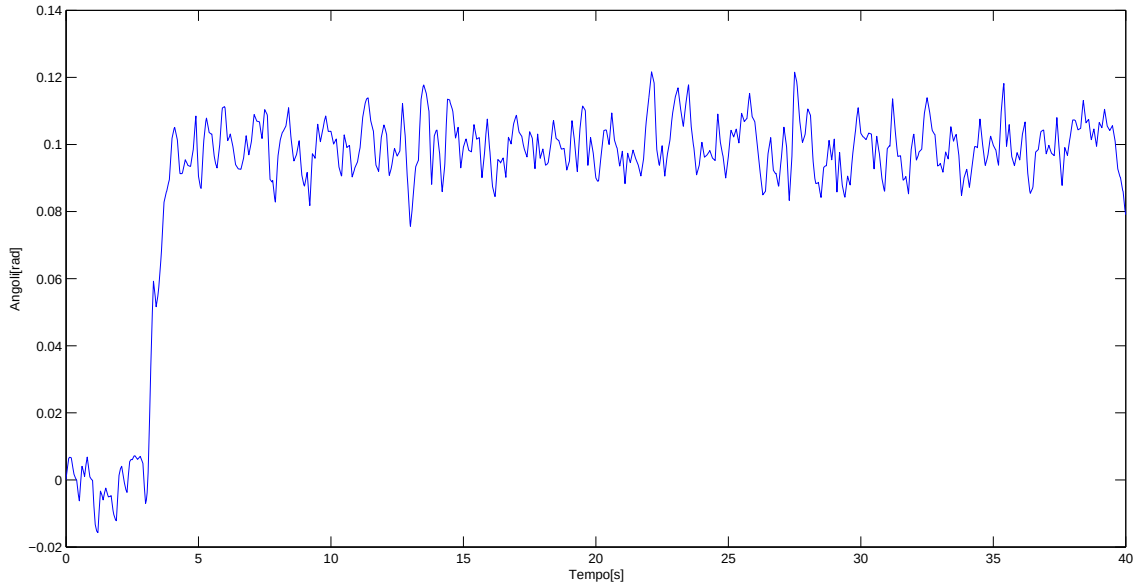


Figura 4.19: Manovra controllo rollio ϕ

per apprezzare le qualitÃ della stima da parte del filtro, si mostra di seguito anche la simulazione dell'andamento dello stato ideale non affetto da rumore del sistema non lineare con lo stesso ingresso di riferimento:

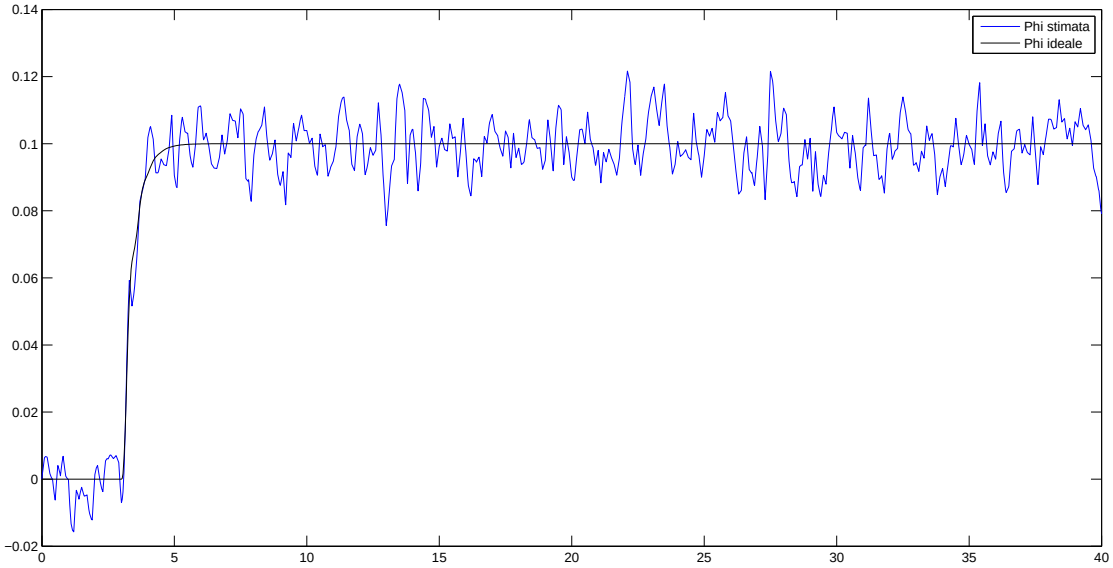


Figura 4.20: Manovra controllo su ϕ con andamento ideale e stimato

Si vuole vedere anche l'andamento nel tempo dell'azione del filtro di Kalman sul sistema non lineare: per questo motivo, qui in basso si fornisce un confronto tra l'andamento dello stato stimato nel sistema linearizzato e in quello non lineare:

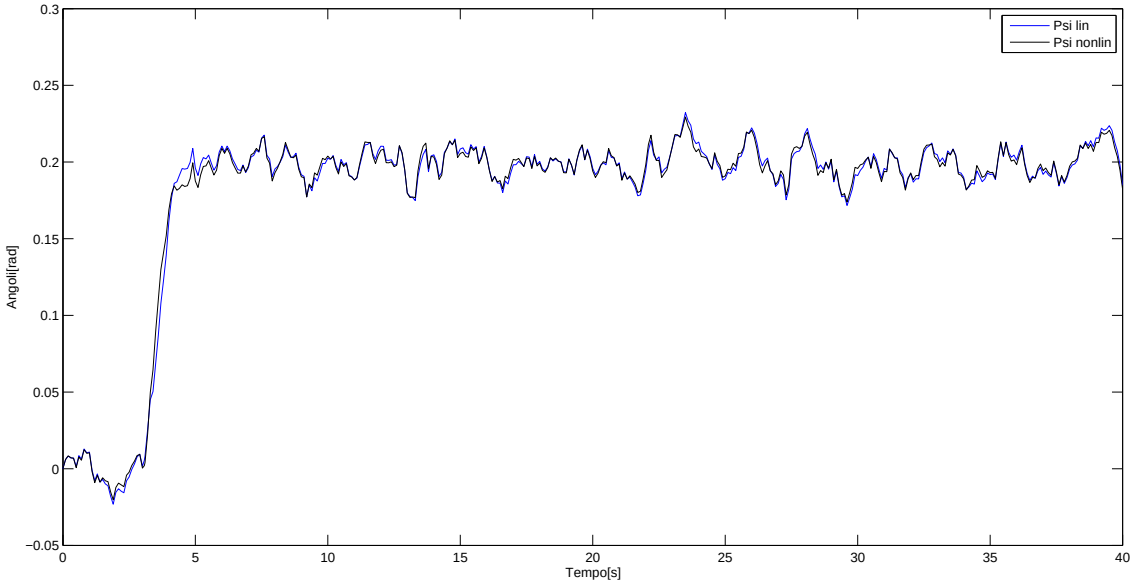


Figura 4.21: Manovra controllo su ϕ: caso linearizzato e non lineare

e si fornisce anche un dettaglio tra i due andamenti:

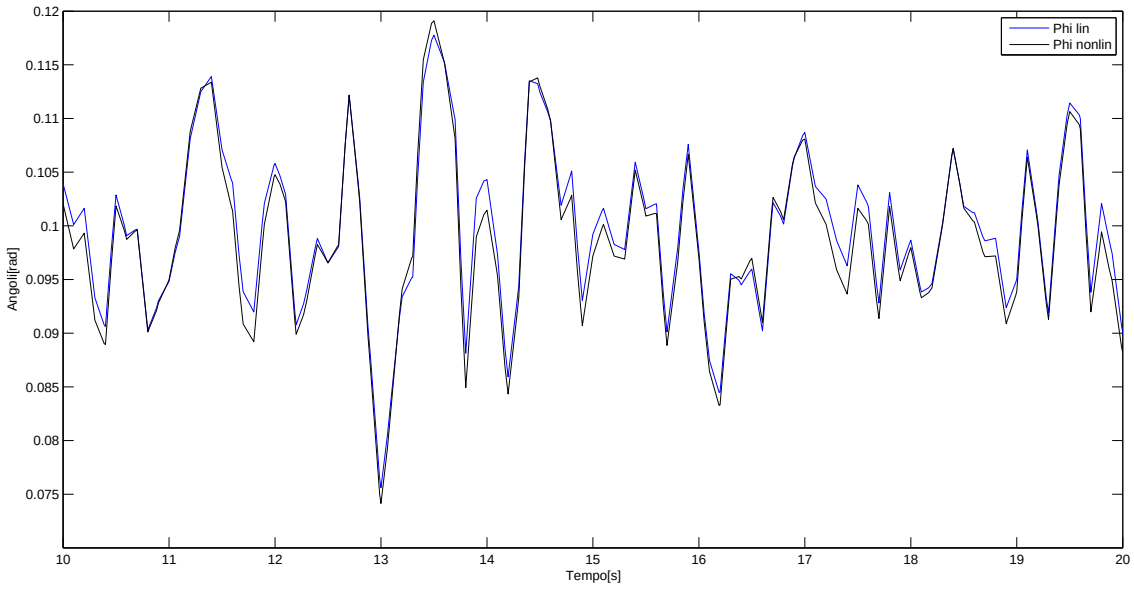


Figura 4.22: Dettaglio del controllo su ϕ:caso linearizzato e non lineare

Con lo stesso ragionamento, si effettua il controllo dell'angolo di beccheggio con stima dello stesso, in risposta ad un ingresso di riferimento di 0.1 radianti posto a 3 secondi:

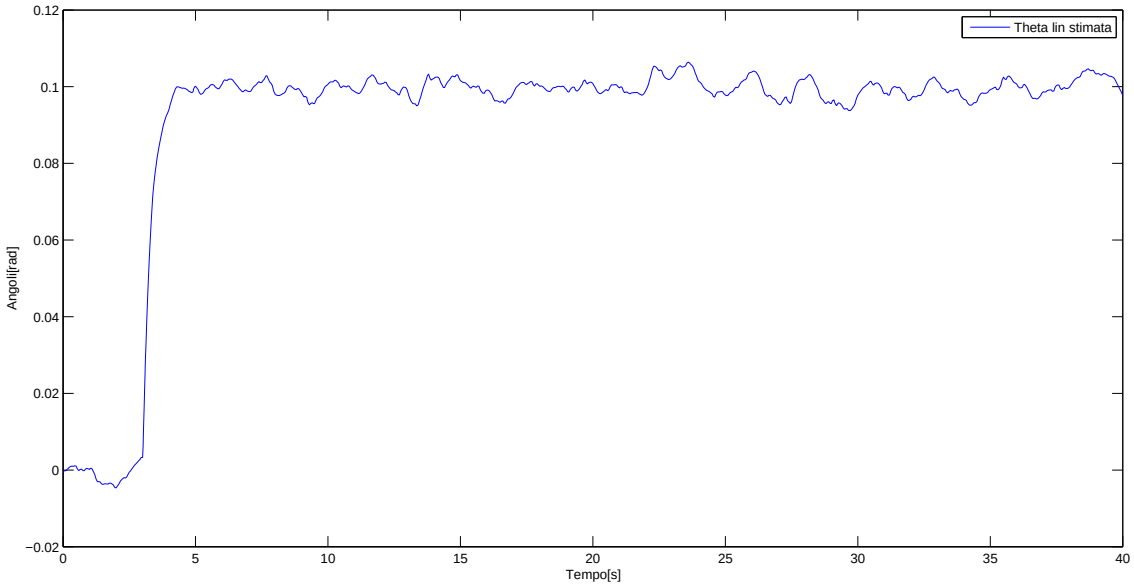


Figura 4.23: Manovra controllo su θ:

mettendolo a confronto con la risposta del sistema non lineare allo stesso ingresso e con stesse condizioni iniziali:

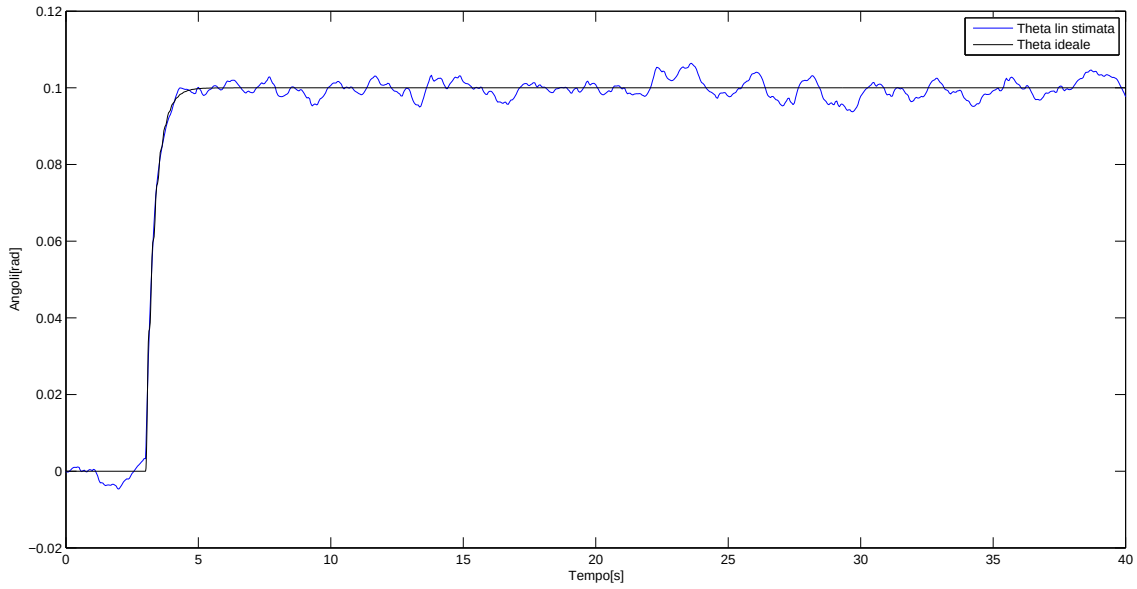


Figura 4.24: Manovra controllo su θ con andamento ideale e stimato

Come fatto per l'angolo di rollio ϕ, anche in questo caso, per l'angolo di beccheggio θ, abbiamo verificato la robustezza del filtro applicandolo al sistema non lineare. Il risultato è il seguente:

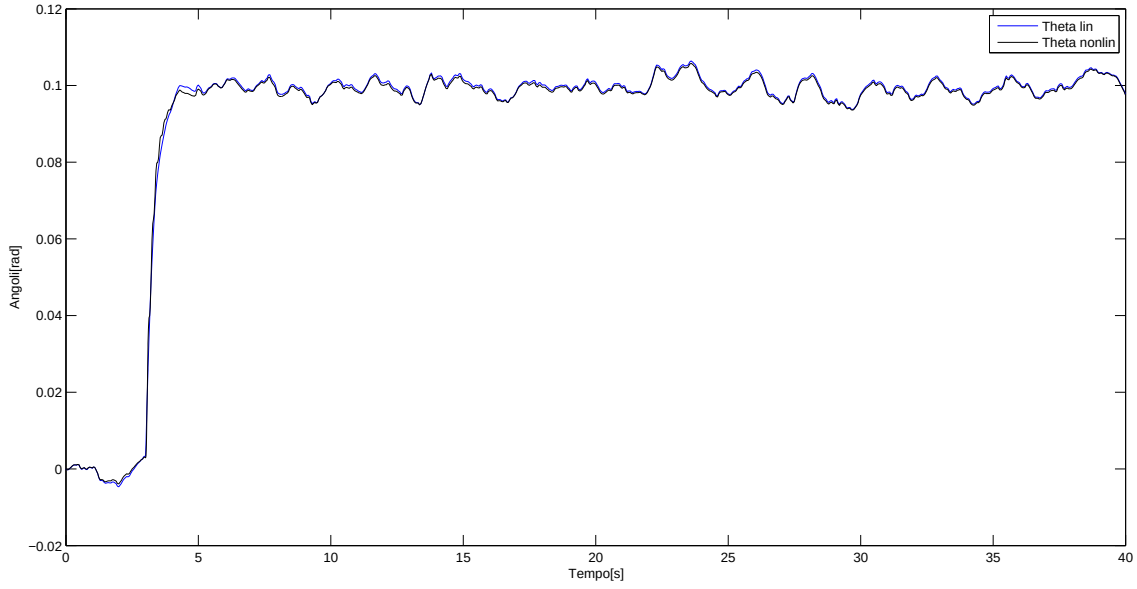


Figura 4.25: Manovra controllo su θ: caso linearizzato e non lineare

con un dettaglio della simulazione:

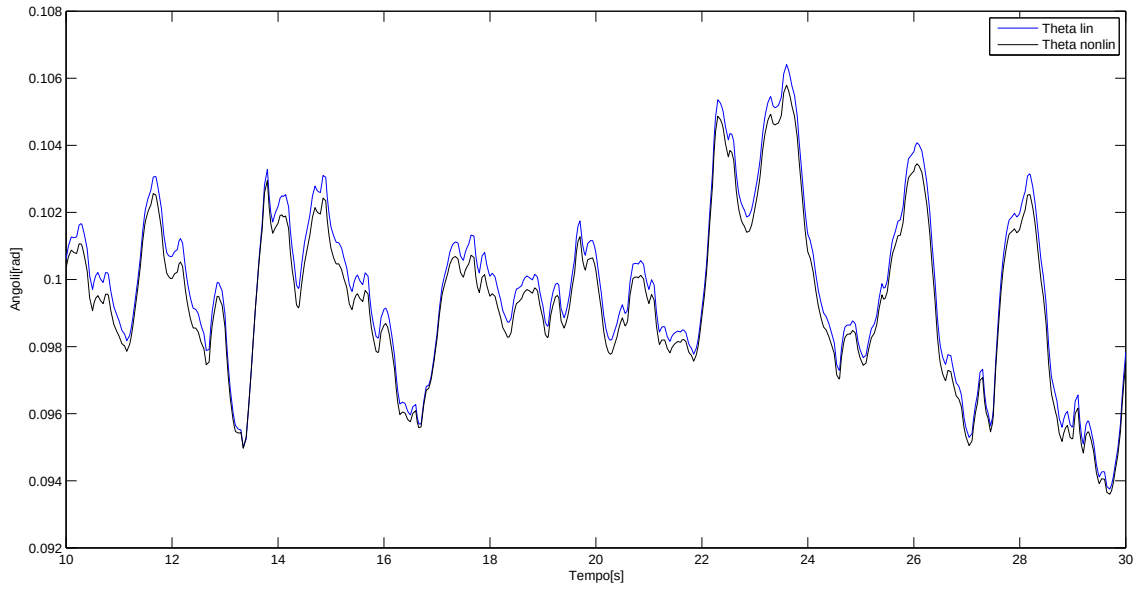


Figura 4.26: Dettaglio del controllo su θ:caso linearizzato e non lineare

Lo stesso ragionamento vale per l'angolo di imbardata, con il seguente risultato:

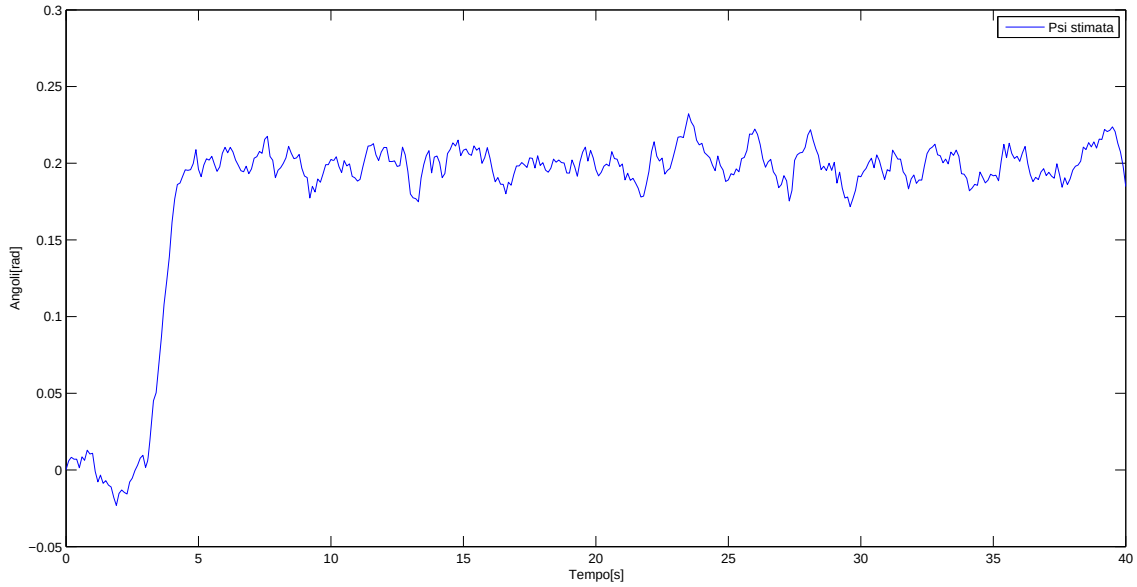


Figura 4.27: Manovra controllo su ψ

messo a confronto con l'andamento dello stato ψ non affetto da rumore:

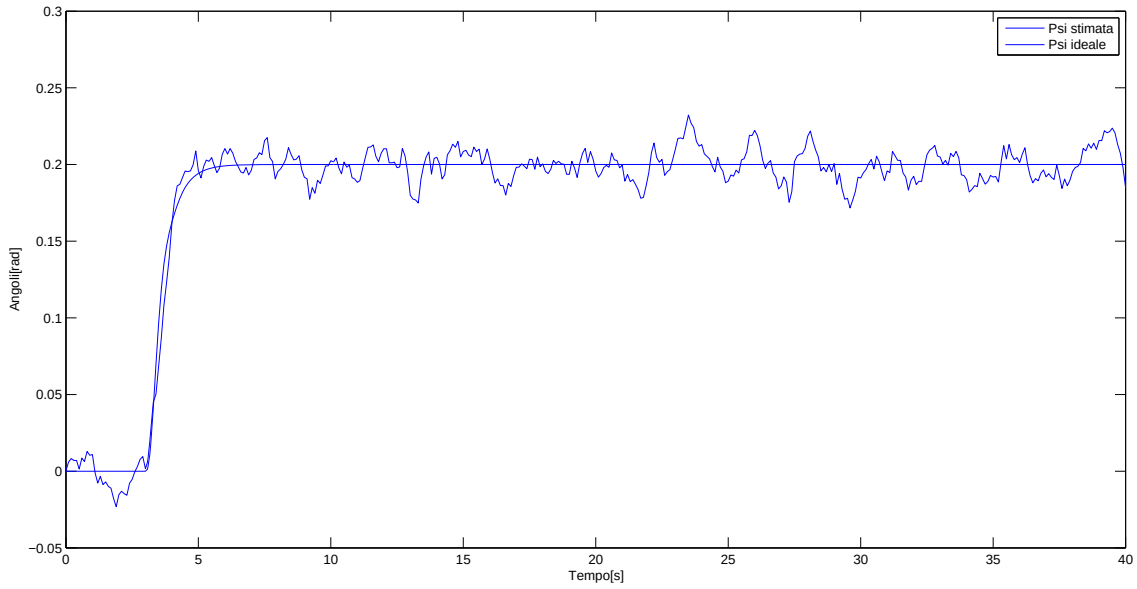


Figura 4.28: Manovra controllo su ψ con andamento ideale e stimato

Anche in questo caso si è verificato l'andamento nel tempo dell'angolo di imbardata ψ stimato nel caso non lineare, mettendolo a confronto con il caso lineare:

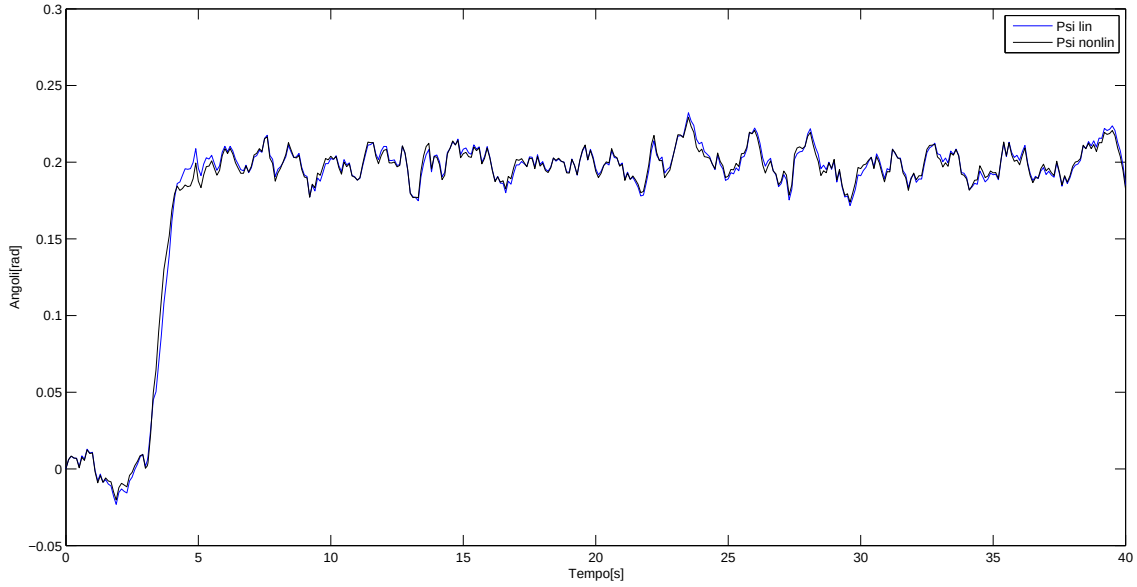


Figura 4.29: Manovra controllo su ψ: caso linearizzato e non lineare

con un dettaglio significativo del confronto:

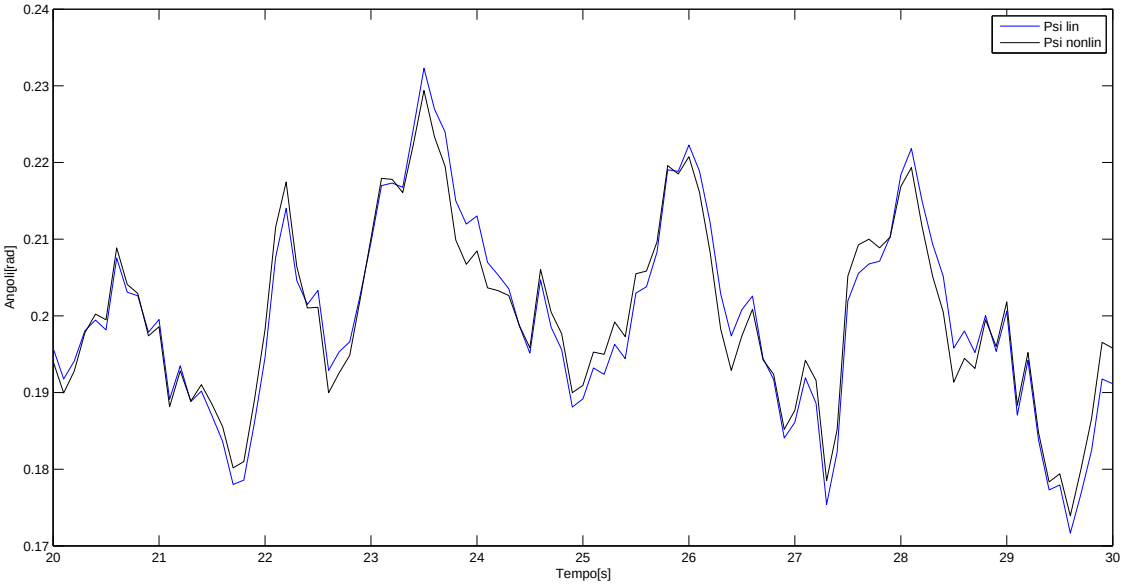


Figura 4.30: Dettaglio del controllo su ψ:caso linearizzato e non lineare

Infine, si forniscono i risultati delle simulazioni sul controllo di altitudine z, partendo dalla risposta del sistema ad un ingresso di riferimento a gradino di 1 metro posto a 10 secondi:

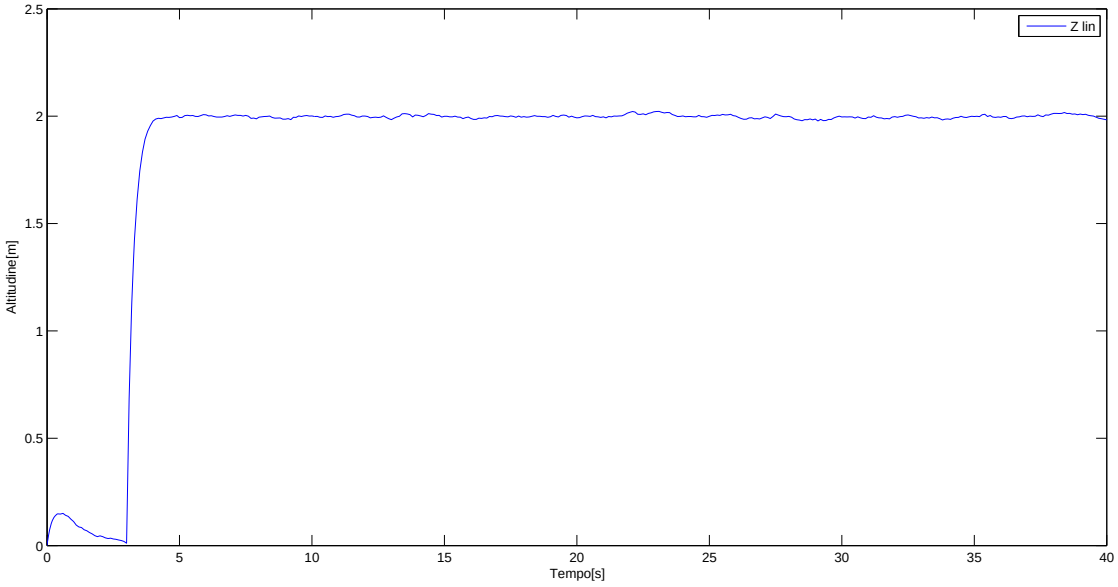


Figura 4.31: Controllo di altitudine z

Si confronta con il caso ideale, di stato non affetto da rumore:

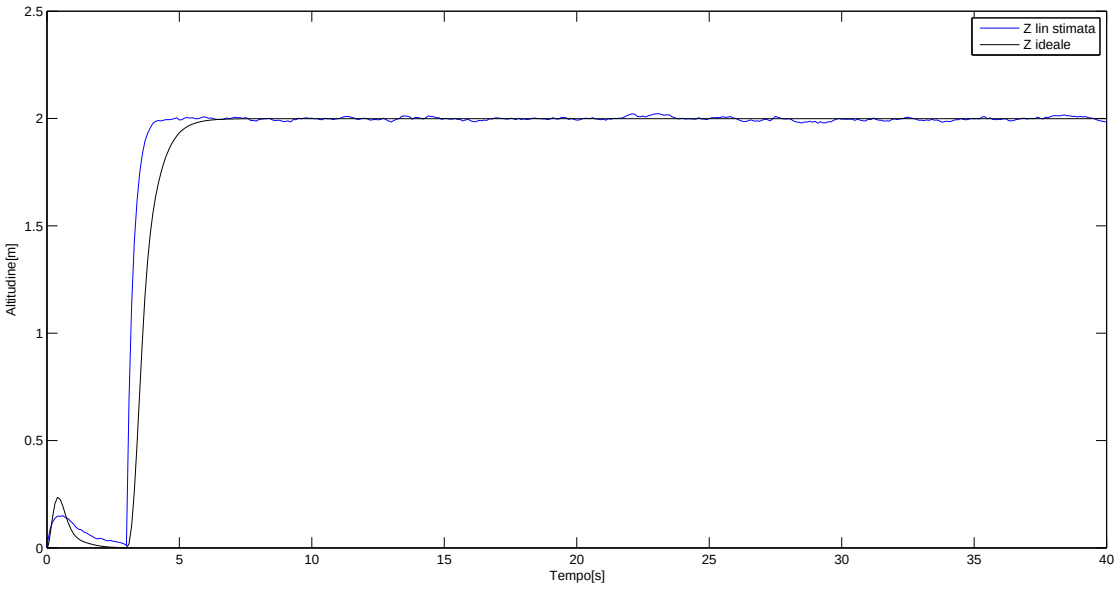


Figura 4.32: Controllo di altitudine z: confronto caso reale e ideale

Nel caso non lineare, viene fornita la risposta del filtro nel seguente confronto con il caso linearizzato:

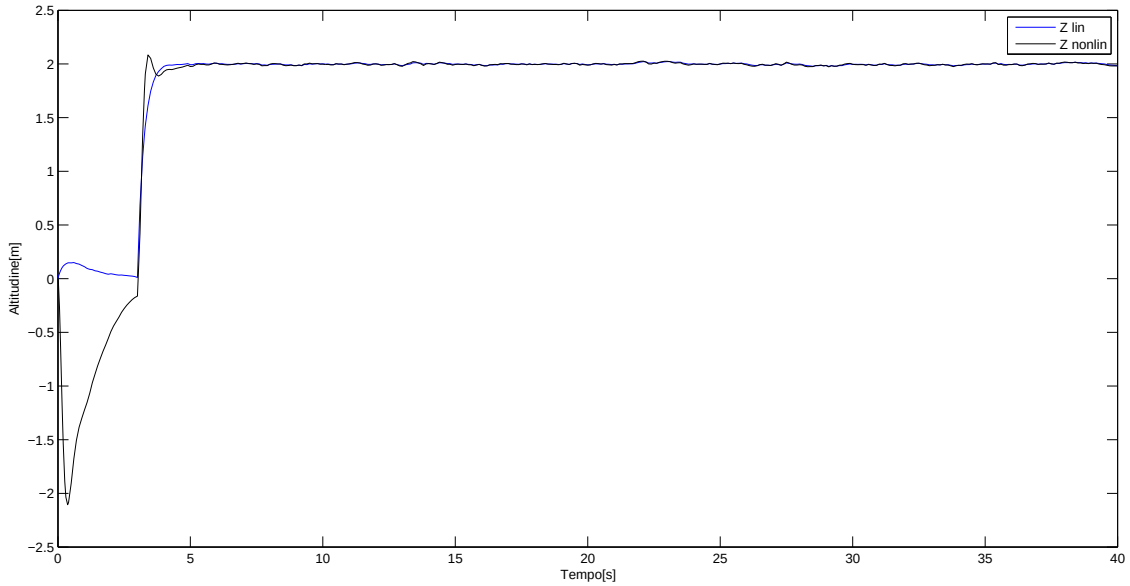


Figura 4.33: Manovra controllo su ψ: caso linearizzato e non lineare

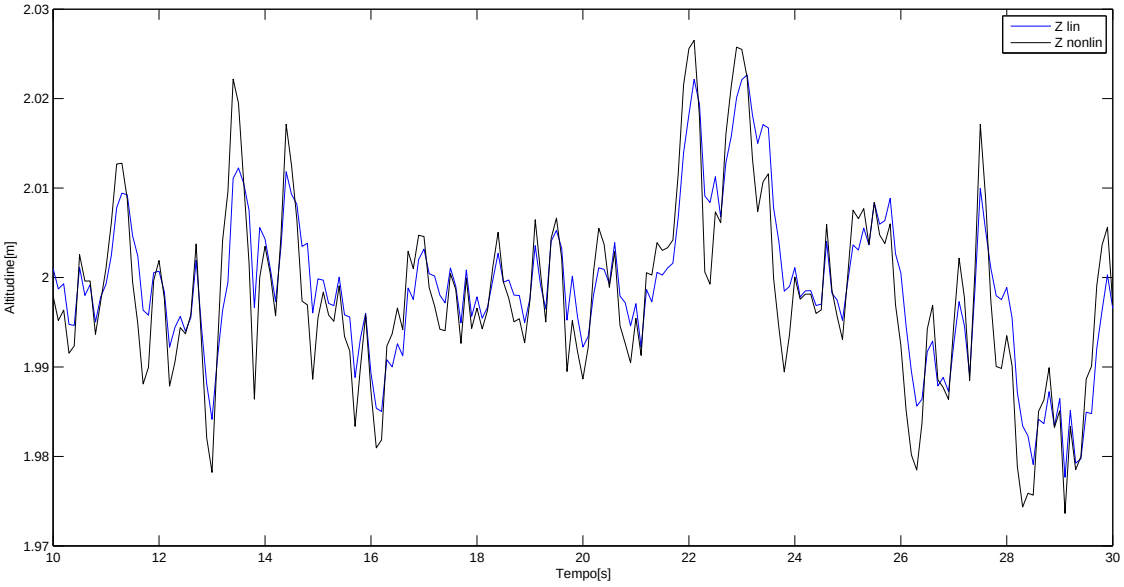


Figura 4.34: Dettaglio del controllo su ψ:caso linearizzato e non lineare

Capitolo 5

Componenti e costruzione di un prototipo di *UAV*

In questo capitlo verrà mostrata la componentisctica e i vari passi susseguitosi durante la realizzazione di un modello reale di un UAV. La costruzione di un prototipo permette di evidenziare, con risparmio di tempo e denaro, gli eventuali errori nella fase di design del controllore. Con la costruzione di un prototipo è possibile anche effettuare una migliore taratura degli eventuali parametri del controllore ed identificare al meglio altri parametri del sistema.

5.1 Descrizione delle componenti

Prima di mostrare le varie componenti utilizzate, e mostrare il processo dietro la loro scelta è opportuno mostrare quali siano i componenti e come essi sono disposti. Dall'immagine che segue infatti, a meno del telaio si ha un idea di come sia il circuito a bordo del quadrirotore oltre a vedere quali siano i componenti installati.

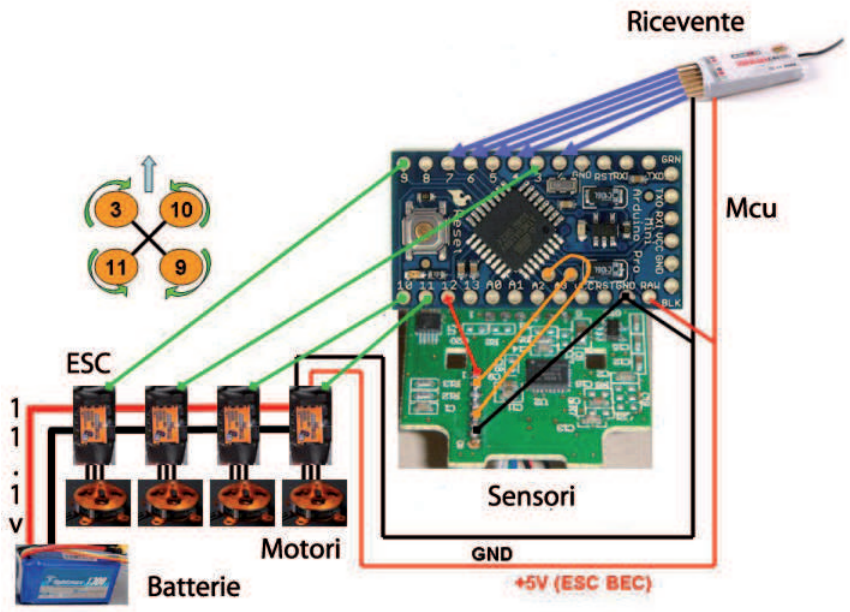


Figura 5.1: Schema riassuntivo dei vari componenti installati sull'UAV

Partendo da in alto a destra si ha in primo luogo la ricevente (Fig. 5.12), questo è un apparato che riceve i segnali di riferimento de un trasmittente (Fig. 5.13) e li trasforma in segnali elettrici (segnali PWM a 50 Hz con duty cycle variabile tra il 5% ed il 10%) che vanno in ingresso alla *Micro controller unit*. La MCU è il vero e proprio controllore del sistema ed è basata su piattaforma Adrunino©. Da questa partono i 4 segnali di controllo verso dei dispositivi elettronici chiamati *Electronic speed Controllers* (Fig. 5.10) che hanno il compito sia di gestire l'alimentazione (che deve esser adattata dagli 11.1 v della batteria (Fig. 5.19) ai $5v$ con cui funzionano le altre apparecchiature) e garantire una sufficiente corrente ai motori raffigurati schematicamente poco più sotto. Si osservi che le linee di alimentazione sono due, entrambe gestite dalle ESC ma aventi la comune sorgente della batteria. Le componenti usate nella costruzione di questo quadrirotore sono stati oggetto di lunga ricerca che ha ci ha impegnato a lungo soprattutto per cercare di avere dei componenti che fossero *low cost* e allo stesso tempo di buona qualità.

5.1.1 Telaio

Il telaio è una struttura meccanica rigida sui cui sono installati tutti i componenti che verranno descritti in seguito. E' la vera e propria ossatura del quadrirotore. La scelta del telaio è stata fatta in funzione principalmente del peso; infatti in questa applicazione è stato necessario cercare di ridurre al minimo tutti i pesi cosi da poter avere un certo *margine* nella scelta della eliche. Tanto minore è la massa, tanto minore è la forza peso e pertanto, con riferimento al capitolo 2, al fine di poter garantire il volo a parità di spinta impressa dalle eliche tanto minore è la forza da vincere in una condizione di volo livellato. Il telaio scelto da una vista in pianta si presenta a forma di croce; all'estremità dei segmenti della croce sono installati gli attuatori, lungo i bracci le ESC e verso il centro si concentra il sistema di alimentazione e l'elettronica di controllo.
Al fine di non entrare nell'inutile complicazione della costruzione del telaio (con relativi annessi problemi come bilanciamento statico e dianmico, scelta dei materiali e garanzia di forma) si è preferito acquistarne uno da un ditta polacca Quadframe [4]. Il telaio scelto [3] pesa circa 215 g ed è realizzato in alluminio e vetroresina G10. E' costruito su più livelli cosi da permettere un facile organizzazione delle componenti ed ha un lunghezza *motore-motore* di 375mm.

Figura 5.2: Il telaio Quad01 scelto per la realizzazione, dopo l'assemblaggio

5.1.2 Motori

Il prototipo costruito è dotato di 4 motori. I motori scelti per questa applicazione sono stati scelti assieme alle eliche al fine di poter raggiungere un spinta sufficiente per permettere il volo al quadricottero. La scelta è ricaduta su un motore tri-fase prodotto dalla cinese E-max che abbinato con le appropriate eliche (par. 5.5) ed Esc (par. 5.10) permette delle buone prestazioni.

Figura 5.3: Il motore $CF2822$ trifase utilizzato

Le caratteristiche di questo motore sono riassunte nella seguente tabella da cui si evince che con le eliche scelte permette una spinta prossima a circa 840g.

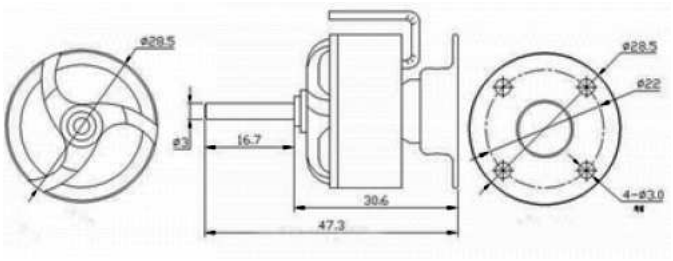


Figura 5.4: Viste del motore $CF2822$ trifase utilizzato

Peso	33 g
Diametro albero	3mm
Massima corrente (60s)	10A
Max Rpm	7700
Rpm/V	1200
Spinta approx. con eliche da 10”	850g

Tabella 5.1: Tabella caratteristiche motore

5.1.3 Eliche

Figura 5.5: Elica EPP1045

Le eliche in un applicazione come questa giocano chiaramente un ruolo chiave. La scelta è stata particolarmente difficile; in letteratura infatti i parametri con cui scegliere le eliche sono numerosi e anche supponendo un buona conoscenza di detti parametri non sempre risultano facili da trovare in commercio i profili voluti. A questo proposito allora con riferimento a [**?**]bbiamo potuto leggere che i principai 3 paramentri sono:

- Coefficienti di spinta c_S
- Coefficiente di potenza c_P
- Raggio r

da questi tre parametri è possible tramite la seguente formula calcolare la spinta di un’elica:

$$S = c_s \frac{4\rho r^4}{\pi^2}\omega^2 \tag{5.1}$$

e la *Potenza* P_p:

$$P_p = c_p \frac{4\rho r^5}{\pi^3}\omega^3 \tag{5.2}$$

Da queste formule si evince chiaramente come la spinta cresca rapidamente con il crescere del raggio dell’elica; dunque un grande raggio dell’elica porta ad avere, a spinta fissata, velocità angolari più basse. Questo evidentemente permette regimi di rotazione più bassi e dunque consumi minori. E’ però allo stesso tempo ovvio che il raggio dell’eliche non possa crescere a dismisura ma debba restare contenuto; con l’elica da noi scelta è il raddio è pari a circa $12.5cm$ In realtà le formule per la spinta e la potenza non tengono conto che il coefficiente di spinta e di potenza non sono costanti[1] ma bensi variano con la seguente formula:

$$J = \frac{u_0}{nD_p} \tag{5.3}$$

[1]si osservi che sono anche formule in forma chiusa che discendono da complicate equazioni differenziali della fluidodinamica

dove u_0 è la velocità del nostro quardirotore, n è la velocità dell'elica espressa in giri al secondo e D_p è il diametro dell'elica.
Tuttavia, al fine di semplificare un poco l'analisi per le eliche, abbiamo trovato in letteratura dei grafici che legano i coefficienti di spinta e di potenza al passo ed al rapporto tra la velocità di avanzamento fratto la velocità angolare. Dall'analisi di questi è ragionevole poter considerare c_S e c_P costanti a basse velocità; questa è infatti la strada che abbiamo seguito noi. Le figure che seguono chiariscono quanto appena espresso:

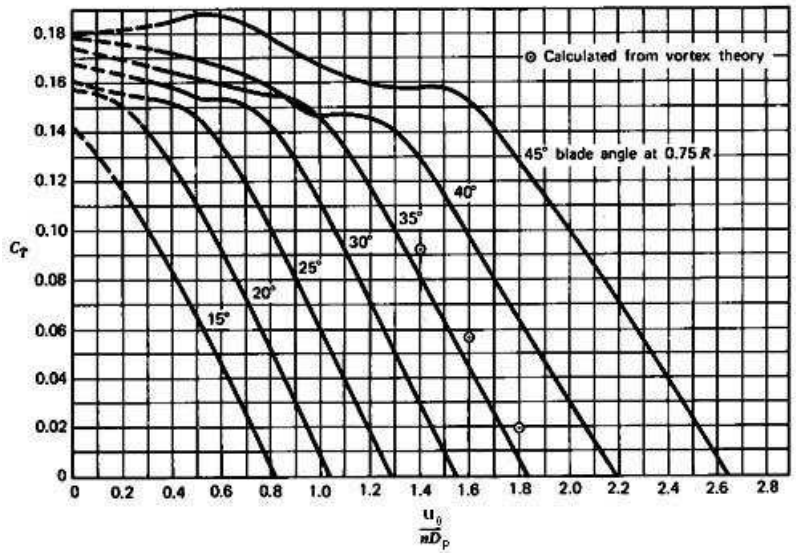


Figura 5.6: Grafico del coefficiente di spinta al variare di J e del passo dell'elica

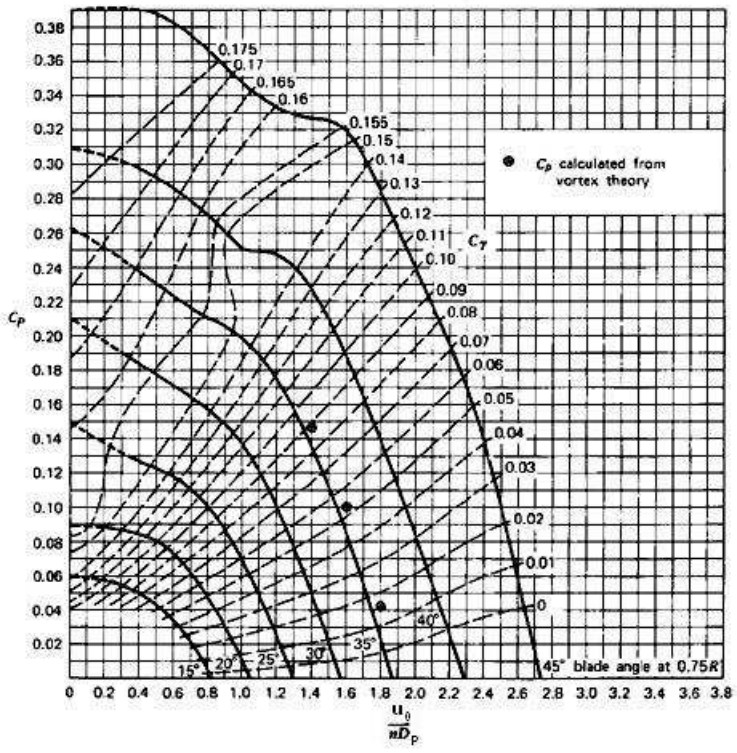


Figura 5.7: Grafico del coefficiente di potenza al variare di J e del passo dell'elica

In virtù di queste considerazioni, analogamente a quanto abbiamo fatto con i motori abbiamo scelto delle eliche facilmente reperibili in commercio: EPP1045 con diametro da $10''$. Una volta acquistate le eliche e alla luce di alcuni dei parametri fondamentali mostrati le prestazioni,abbiamo provveduto a trovarne i parametri. Per quanto visto nelle fig. 5.6 e fig. 5.7 abbiamo pertanto supposto i coefficienti costanti essendo le velocità basse nel nostro modello. Al fine di trovare dei valori plausibili per i coefficienti c_S e c_P abbiamo trovato una valida risorsa in rete da cui siamo riusciti a calcolarli la media di c_S e c_P da alcuni dati tabellati ottenendo i seguenti risultati:

$$c_S = 0.1154 \qquad c_P = 0.0743 \tag{5.4}$$

Dunque trovati questi valori e supponendo il peso del prototipo sia pari ad $1Kg$ ogni gruppo di propulsione motore-elica deve permettere una spinta di circa 2.45. Questo valore è dato dal fatto che ogni motore deve vincere un quarto della forza peso a cui è soggetto l'intero prototipo; nell'ipotesi fatta di peso pari a $1Kg$ a cui corrisponde una forza peso di $9.81N$ si ha che $\frac{9.81}{4} = 2.45$. Questo valore di spinta è chiaramente il valore minimo di spinta che si deve poter raggiungere per una condizione di volo livellato. Il valore di spinta minima che deve essere soddisfatto è stato quindi estratto dal grafico di che segue che rappresenta la spinta S in funzione della velocità angolare con $\rho = 1.18$.

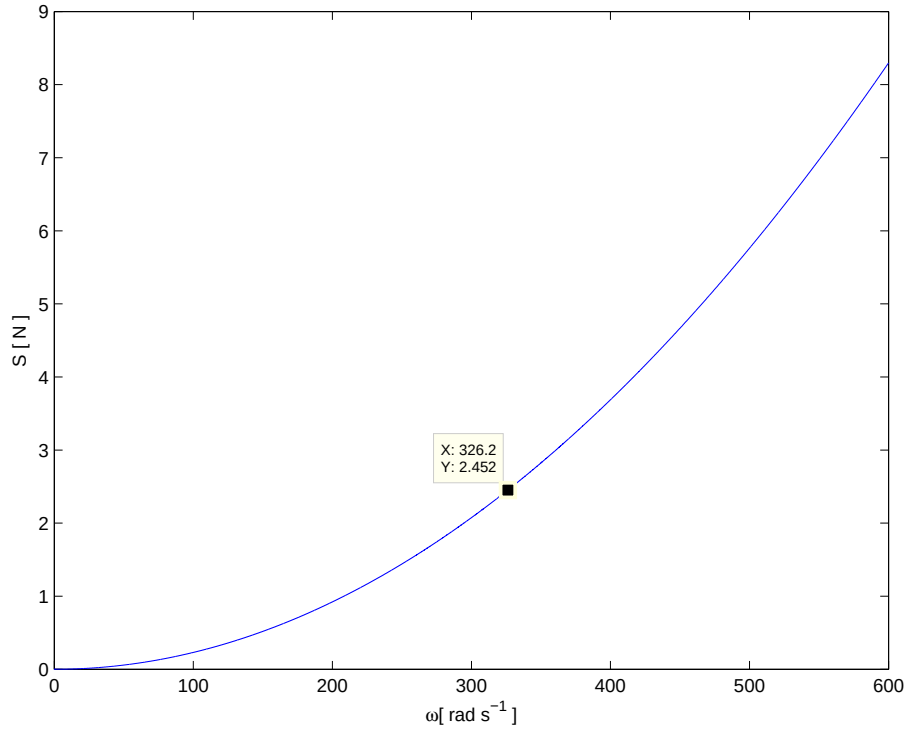


Figura 5.8: Spinta in funzione della velocità angolare

La velocità angolare per alzarsi in volo è quindi pari a $323rad \quad s^{-1}$ che espressa in termini di r.p.m è pari a circa 3085; velocità facilmente raggiungibile dai motori.
Altro paramento a proposito della scelta delle eliche è la potenza che è graficamente mostrata nella seguente immagine il valore di interesse è quello proprio alla velocità angolare che permette di vincere la forza peso. In questo caso è pari a circa $27W$

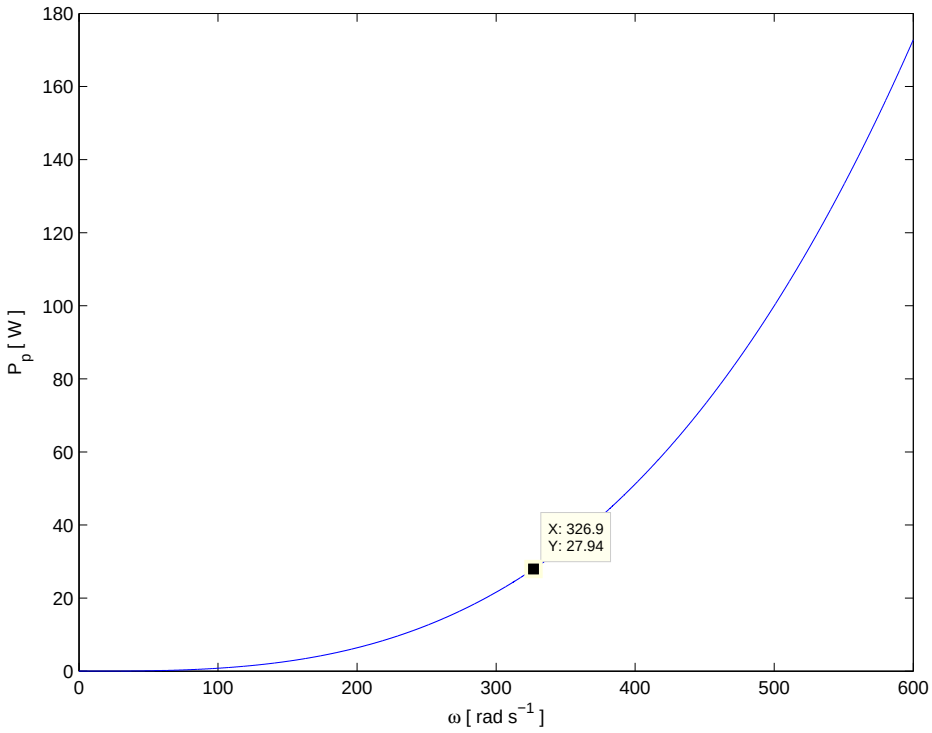


Figura 5.9: Potenza in funzione della velocità angolare

5.1.4 Electric speed controllers

Le *Electronic speed controller* sono dei dispositivi elettronici atti non solo a convertite la corrente monofase dei segnali di controllo in corrente trifase, ma peremttono anche di erogare un quantità di corrente sufficiente agli attuatori. Infatti tipicamente i microcontrollori non possono erogare più di circa $20ma$ complessivamente. Il loro compito è pertanto simile a quello di un ponte ad H, con la differenza che i più moderni hanno schemi più complessi per una maggiore efficienza e sicurezza. Infatti le ESC scelte hanno la capacità di tararsi attorno al segnale di zero e permettono di distribuire l'alimentazione anche al sistema di ricezione di segnali radio.

Figura 5.10: E.s.c. da 20*A* complessivi con sistema *BEC*

5.1.5 Sistema Radio

Il sistema di radio scelto della Turnigy è stato scelto in funzione della possibilità di usare una comunicazione radio a 2.4Ghz a *Spread Spectrum*, o espansione spettrale, con una quasi totale assenza di interferenza, differentemente da quanto accade nei sistemi a trasmissione FM. In questo tipo di trasmissione in fatti la banda del segnale da trasmettere è considerevolmente più bassa rispetto a quella della banda in cui è *possible* trasmettere e quindi si fa variare il canale di trasmissione all'interno della banda di trasmissione in maniera random al fine di essere meno sensibile alle interferenze. Questo tipo di comunicazione nasce durante la seconda guerra mondiale al fine di evitare il cosiddetto *Jamming [5]* in cui al fine di interrompere le trasmissioni, che avvenivano a fissata frequenza, si mandava su quella frequenza un forte disturbo. Saltando dunque tra i vari canali disponibili all'interno della banda portante se anche uno dei canali è molto disturbato la probabilità di avere interfere diventa, detto m il numero di canali, è pari a $\frac{1}{m}$

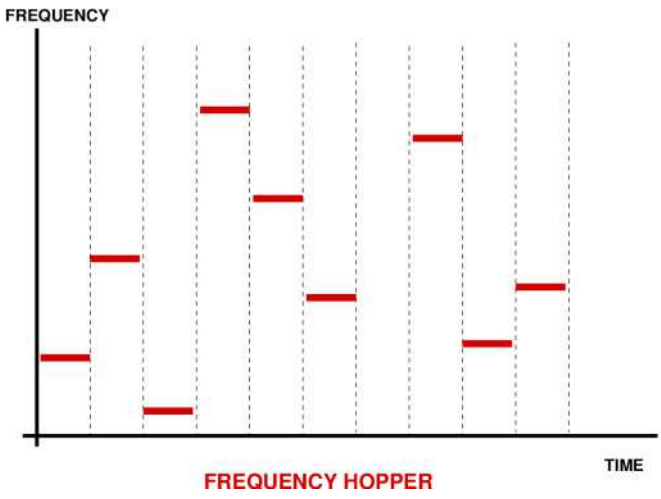


Figura 5.11: Schema frequency hopping Time vs Frequency

Il sistema radio scelto questione dispone di 9 canali differenti[2], ad ognuno dei quali è possibile associare un attuatore (motore o servocomando che sia).

Figura 5.12: Apparato trasmittente del sistema radio Turnigy a 9 canali

[2]si osservi che per canale qui intende canale su cui è possibile dare un comando e *non* un canale di trasmissione

Figura 5.13: Apparato trasmittente del sistema radio Turnigy a 9 canali

5.1.6 Microcontrollore

Visto il ridotto ingombro del quadrirotore e in funzione della semplicità di programmazione abbiamo scelto per questa applicazione di controllo un Microcontrollore (anche detta Micro Controller Unit M.C.U) Arduino Mini Pro. Questo è infatti adatto in virtù non solo dei numerosi pin di input ed output di cui dispone ma anche grazie alle numerose librerie preesistenti a riguardo di: comunicazione seriale, lettura dei segnali di riferimento in ingresso dalla ricevente radio, gestione dei servomotori.
In virtù di quanto può essere letto in questo paragrafo ed in relazione a quanto già esposto nel paragrafo di identificazione dei motori, tutti i circuiti a meno della MCU funzionano con segnali a frequenza pari a 50Hz. La stessa MCU funziona ad un frequenza (del loop di controllo) pari a 490Hz; riuscire pertanto a sincronizzare le frequenze della MCU e dell'apparto Hardware non è un compito facile e soprattutto esula dalle finalità di questa tesina. Per questo motivo, oltre agli altri già esposti, la nostra scelta è ricaduta sull'MCU Arduino Mini Pro dove gestire problemi relativi a diverse frequenze è affidato a strutture software già esistenti di facile gestione.
L'Arduino Pro mini dispone utlizza un processore ATMega-328 e usa supporta un codice C-like di cui abbiamo fatto uso.

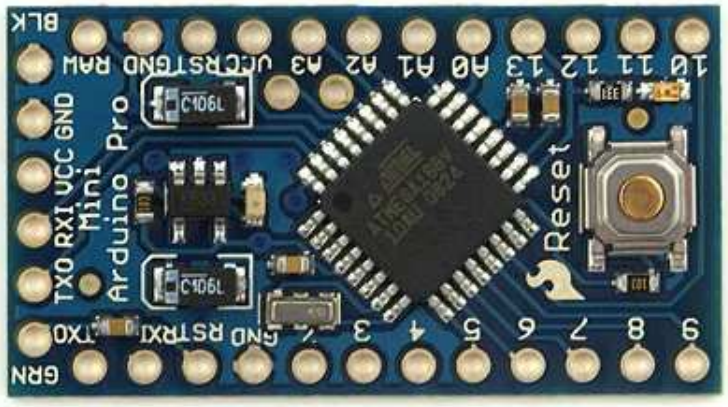


Figura 5.14: MCU Arduino Pro Mini

Memoria

L'Atmega-328 ha 32 KB, di cui 2 KB utilizzati per il bootloader, 2 KB di SRAM e 1 KB di EEPROM.

Comunicazione

L' Arduino Pro Mini ha un certo numero di strutture per la comunicazione con un computer, con un altro Arduino, o con altri microcontrollori. L'Atmega-328 fornisce una comunicazione seriale UART TTL (5V) che é disponibile sui pin digitali 0 (RX) e 1 (TX). Il software predisposto alla comunicazione include un monitor Arduino seriale che consente di inviare alla scheda Arduino semplici dati testuali. I led RX e TX sulla scheda lampeggiano quando i dati vengono trasmessi sia tramite il chip FTDI, sia tramite una connessione USB con il computer; ciò non accade per la comunicazione seriale. L'Atmega-328 dispone anche di un supporto I^2C [7](che viene usato per la comunicazione con i sensori) e di una comunicazione SPI. Il software include una libreria Arduino Wire per semplificare l'utilizzo del I2C bus.

5.1.7 Programmatore FTDI

Per poter scaricare il software scritto sulla MCU via USB è stato necessario acquistare un programmatore esterno. Questo non è necessario nel caso in cui la board con la MCU già disponga di un programmatore integrato e dunque di una porta intergrata; tuttavia viste le ridotte dimensioni della MCU l'Arduino Mini pro questa non dispone di un programmatore a bordo. Pertanto abbiamo utilizzato un interfaccia FTDI che servisse al compito descritto; l'interfaccia da noi scelta è la più diffusa che si trova e dopo un brave ricerca sembra essere la meglio funzionante.

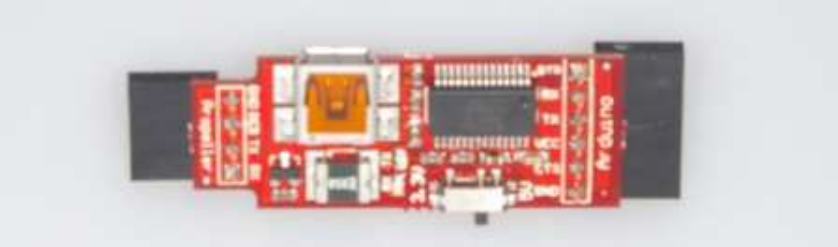


Figura 5.15: Interfaccia FTDI per la comunicazione USB

5.1.8 Sensori

I sensori scelti per il quadrirotore sono due, uno accelerometrico e un giroscopico. Entrambi i sensori comunicano con la MCU a mezzo di un protocollo detto I^2C [7]; un protocollo digitale tra circuiti integrati le cui API sono facilmente utilizzabili e ben note. LA frequenza di comunicazione è stata fissata a 100kHz sebbene sia possibile permettere comunicazioni a 400kHz. I sensori sono stati presi da *controller* per piattaforma *Wii*.[3].
La scelta di utilizzare dei sensori già installati in due apparecchi preesistenti è stata fatta in funzione del basso costo a cui è stato possibile trovare un controller di sottomarca, ed in funzione delle prestazioni di questi lette in un pagina web. Oltre tutto spesso i sensori si trovano in forma di circuiti integrati senza piedinatura per la comunicazione esterna e quindi, avendo intuito che i sensori delle console di videogiochi avevano per ogni sensore anche l'elettronica di comunicazione, ci siamo orientati in questa direzione.

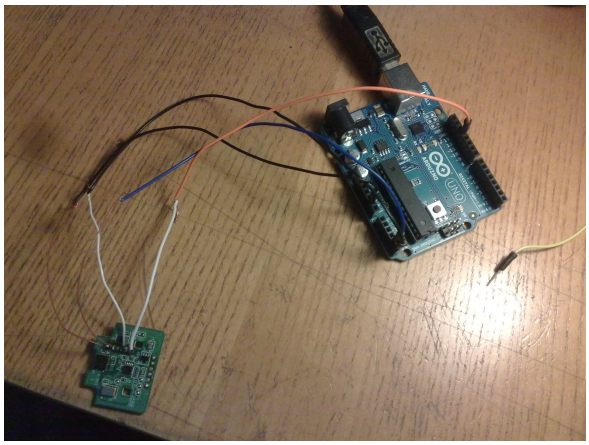
5.1.9 Giroscopio

Il sensore giroscopico a tre assi, a meno della board su cui installato, è prodotto da [2] ed è progettato in tecnologia MEMS [8]. Il modello è chiamato IDG-600 le cui caratteristiche sono riportate nella seguente tabella:

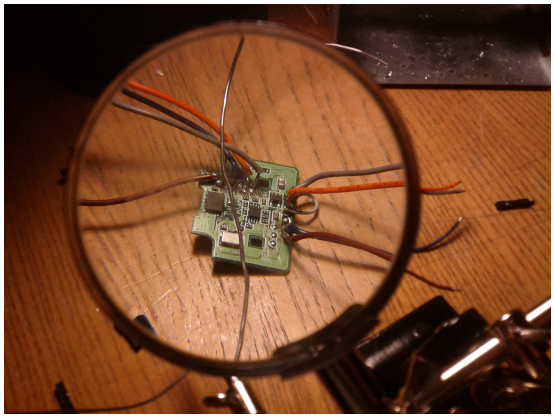
[3]una moderna piattaforma di videogiochi

Fondo scala	±500 a 2000 °/sec
Nonlineartà	< 0.1% del F.S.
Sensitività tra gli assi	<1 %
Banda passante	140Hz
Tensione input	6.0 ± 0.3 V
Peso	0.14g
Massima accelerazione tollerabile senza alimentazione	>10000 g per 0.3ms
Temperatura operativa	-20 to +85 °C

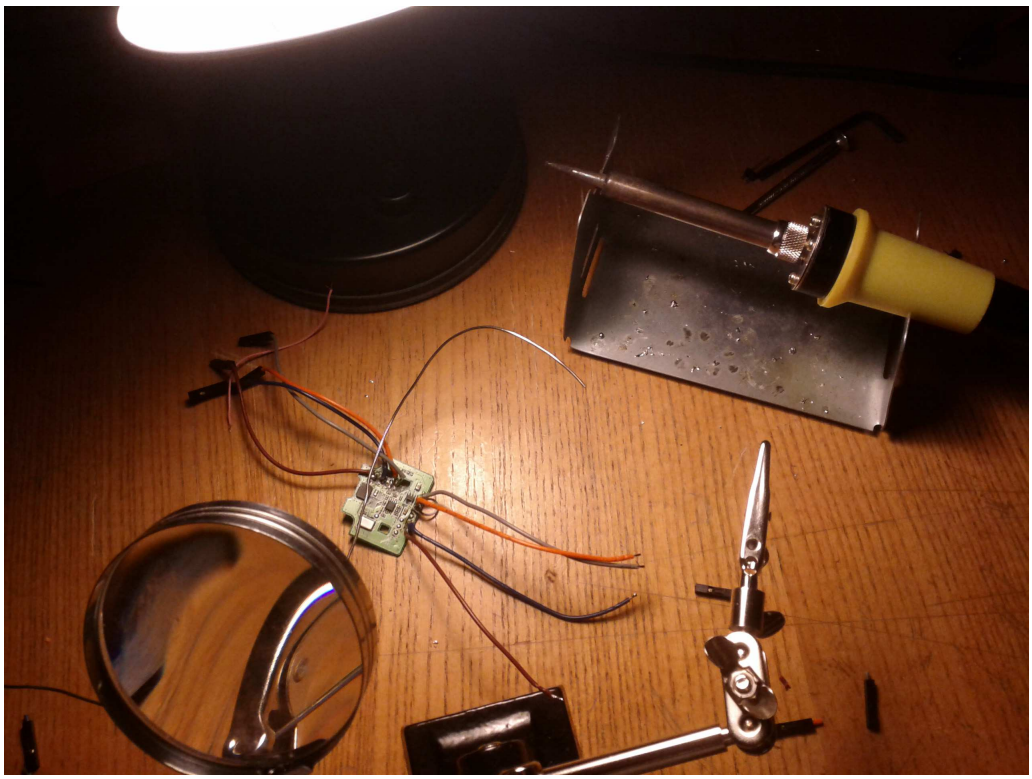
Tabella 5.2: Tabella specifiche IDG-600



(a)



(b)



(c)

5.1.10 Accelerometro

Come già accennato il sensore accelerometrico (a tre assi) è stato preso da un *pad* di una console. Dopo avere appositamente smontato il pad e dopo un breve ricerca su internet abbiamo evinto che il sensore è prodotto dalla ST Mircoelectronics ed è il modello LIS3L02AL. Le specifiche sono riassunte nella seguente tabella:

Fondo scala	$\pm 2g$
Sensistività	$\frac{V_{dd}}{5} - 10\%$
Nonlinearità	tra 1.5% e 1.7% del F.S
Temperatura operativa	tra -40° e 85°
Peso	0.08 g
Tensione alimentazione ($V_d d$)	tra 3V e 47V

Tabella 5.3: Tabella specifiche LIS3L02AL

Anche nel caso del sensore accelerometrico abbiamo abbiamo preferito prenderlo con dell'elettronica attorno che permettesse una facile gestione della comunicazione con protocollo I^2C.

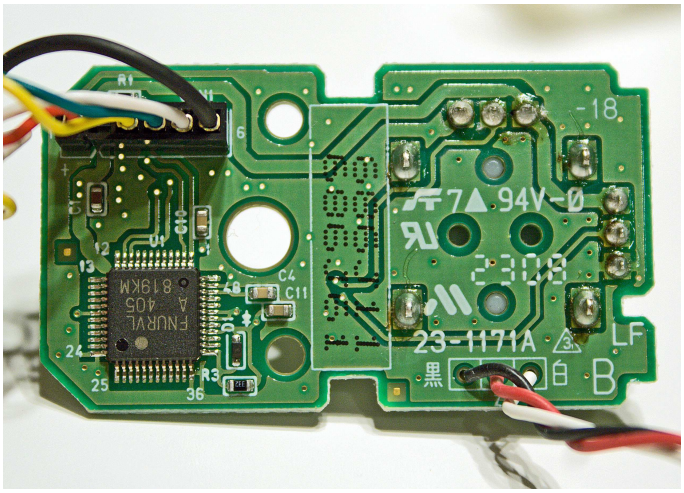


Figura 5.16: Il sensore Accelerometrico

I due sensori prima di essere interfacciati alla MCU si possono vedere nella Fig.5.17.

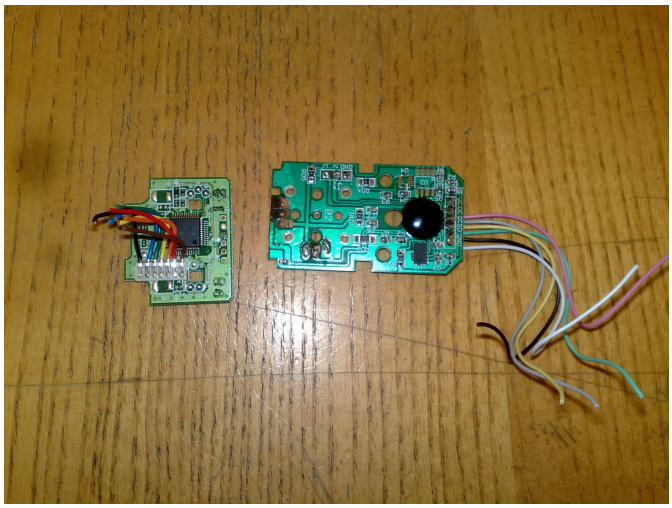


Figura 5.17: Sensore accelerometrico e giroscopico

5.1.11 Sistema di alimentazione e di distribuzione dei segnali di controllo

Al fine di poter avere un collegamento 'pulito' tra i vari componenti abbiamo pensato di realizzare sulla base delle nostre conoscenze un PCB[4] raffigurato nel seguito.

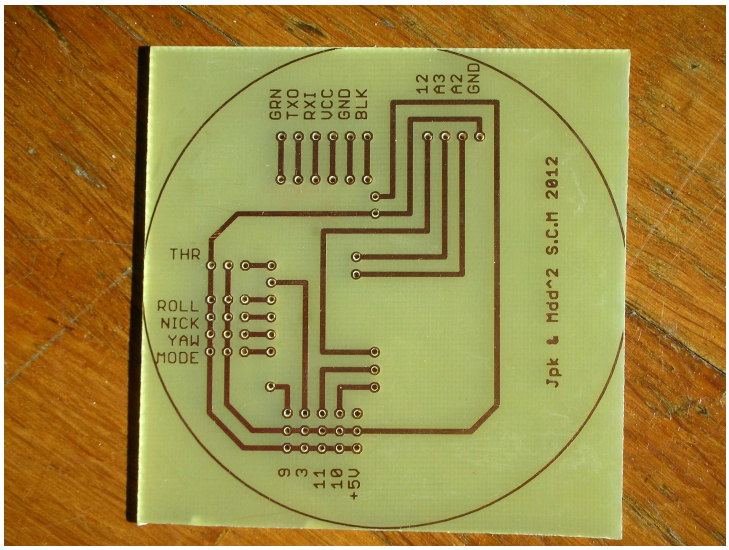


Figura 5.18: Piastra PCB autocostruita. Da notare la particolare serigrafia

La board mostrata permette di effettuare le connessioni e di posizionare i sensori. Nel prossimo paragrafo verrà mostrato come essa è stata realizzata.

5.1.12 Batterie e carica batterie

La batteria scelta è un batteria agli Ioni di litio (a 3 celle ognuna delle quali a $3.7v$ connesse in serie) a $11.1v$ con una capacità di circa 2200 mah che alimenta sia i motori che l'elettronica di bordo. Sebbene quasi tutti i sistemi a bordo del quadrirotore funzionino con tensione comprese tra i $4v$ e $6v$, grazie ad un opportuno collegamento delle E.S.C, come verrà mostrato in seguito,

[4]Printed Circuit Board, ovvero circuito stampato

è possibile alimentare tutti i componenti adattando opportunamente le tensioni. Il motivo per cui si è scelto questa batteria è l'elevata capacità ed il ridotto peso.

Figura 5.19: Batteria al Litio a celle da 11.1v

Il carica batterie scelto è un caricabatterie che permette una carica completamente automatica di un pacco batteria agli ioni di litio. Tra le funzioni che ci interessavano volevamo che vi fosse la possiblità di poter caricare in maniera bilanciata le 3 celle del pacco batteria; come mostra un immagine che segue questo è possibile tramite un apporta porta.

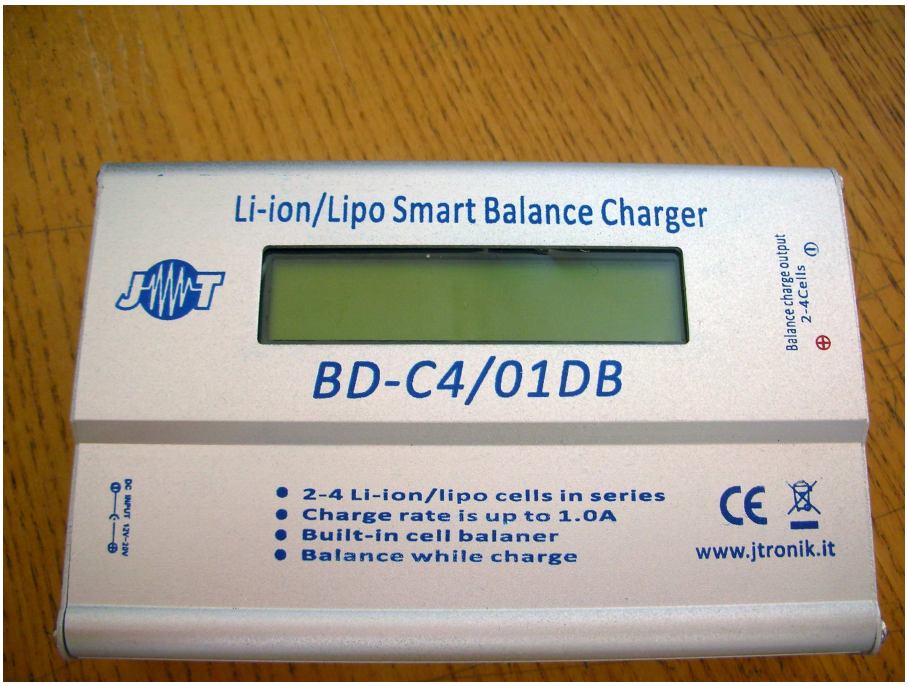


Figura 5.20: Carica batterie Jtronik per la carica delle batterie

5.2 Costruzione del prototipo di Quadrirotore

5.2.1 Sviluppo e costruzione del sistema per la distribuzione dei segnali alimentazione e di controllo

Per la distribuzione dei segnali di controllo e l'instradamento dei segnali da parte del sistema radio a bordo del nostro UAV abbiamo pensato di costruire un PCB che avesse delle piste in

rame. L'uso di un PCB ci ha molto semplificato la gestione delle numerose connessioni tra le varie componenti a bordo del prototipo; abbiamo infatti per un breve periodo ipotizzato l'uso di un piastra mille fori ma dopo alcune prove abbiamo comunque optato per il PCB nonostante la maggiore difficoltà nella realizzazione.
La costruzione di un PCB avviene a mezzo di foto incisione ed attacchi acidi ad una piastra ricoperta di rame. La piastra, detta anche basetta, prima dell'incisione e degli attacchi acidi appare completamente ricoperta in rame, una esposizione di questa alla luce UV, con opportune zone di ombra create utilizzando un disegno appositamente realizzato, indebolisce alcuni legami molecolari e pertanto dopo avere immerso la basetta in un acido il rame esposto ai raggi UV si scioglie lasciando inalterata la parte non esposta ai raggi.
Per prima cosa allora abbiamo realizzato il disegno da incidere sulla basetta. A questo scopo abbiamo fatto uso di un programma ben noto nell'ambito dell'elettronica chiamato Eagle-CAD [6]. Come si puo' vedere dopo avere scelto i pin di input ed output abbiamo creato le connessioni (o piste) ed abbiamo creato appositi *pads*, ovvero zone di rame allargate lungo una pista, su cui posizionare dei piedini. L'immagine che segue mostra la fase conclusiva del disegno che siamo andati ad incidere sulla piastra.

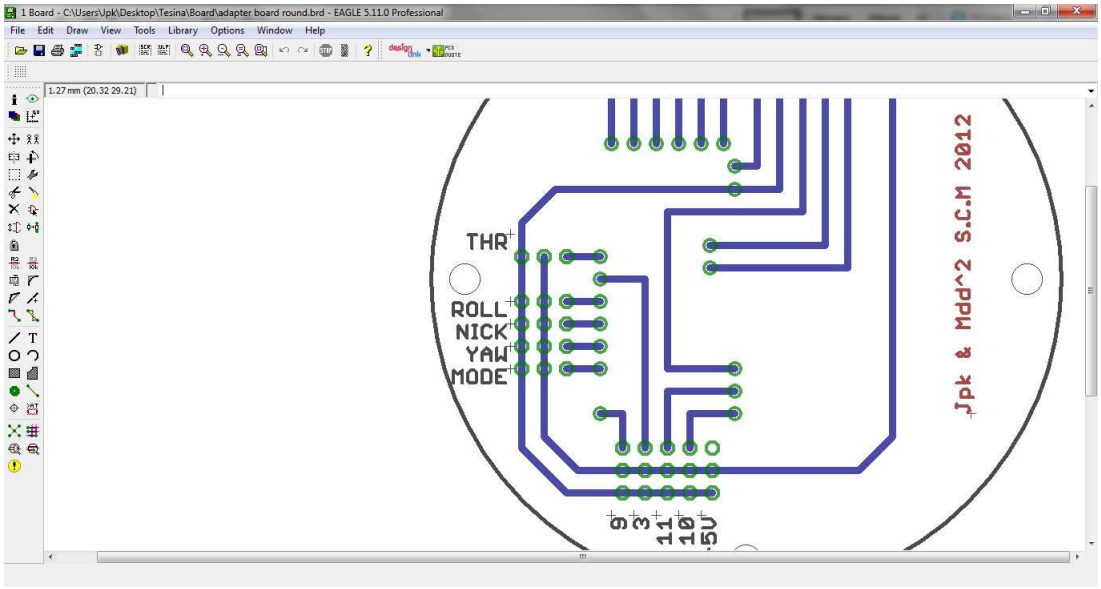
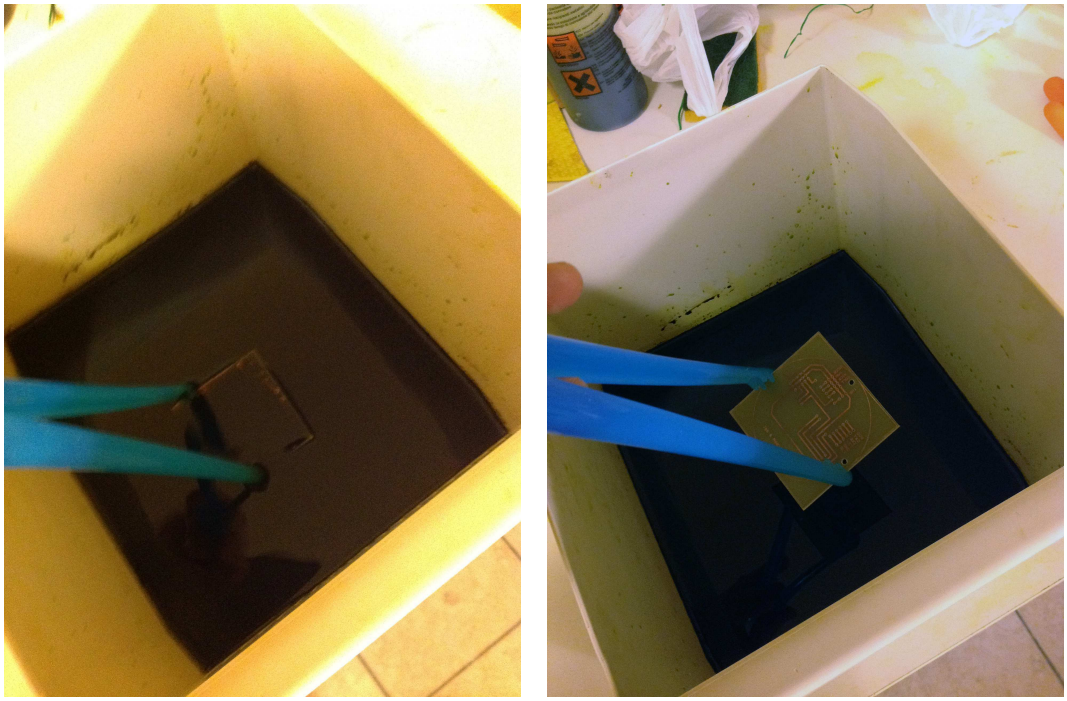


Figura 5.21: Fase di disegno della board nell'ambiente EAGLE-CAD, da notare il particolare della serigrafia

Dopo avere terminato la fase di disegno abbiamo iniziato la fase di foto incisione sulla basetta. Per questa fase, dopo avere stampato su carta il disegno realizzato, è stato utilizzato un macchinario chiamato *bromografo*; questo non è altro che una scatola in sui sono installate della lampade UV ed un temporizzatore cosi da spegnere le lampade dopo il tempo necessario (in questo caso dopo circa 3 minuti). Il bromografo è mostrato nella seguente immagine:

Dopo quindi avere impresso il disegno precedentemente progettato sulla basetta con i raggi UV siamo passati ad immerge il PCB in un acido cosi da rimuovere le zone di rame esposte ai raggi UV. Questa fase ha richiesto qualche minuto ed è stata fatta a temperatura maggiore della temperatura ambiente



(a) (b)

Figura 5.22: Schema identificazione motori

L'ultima fase prima di poter saldare le connessione è stata quella di foratura del PCB effettuata con un trapano con punta da $0.01mm$ come mostrato in Fig. 5.23

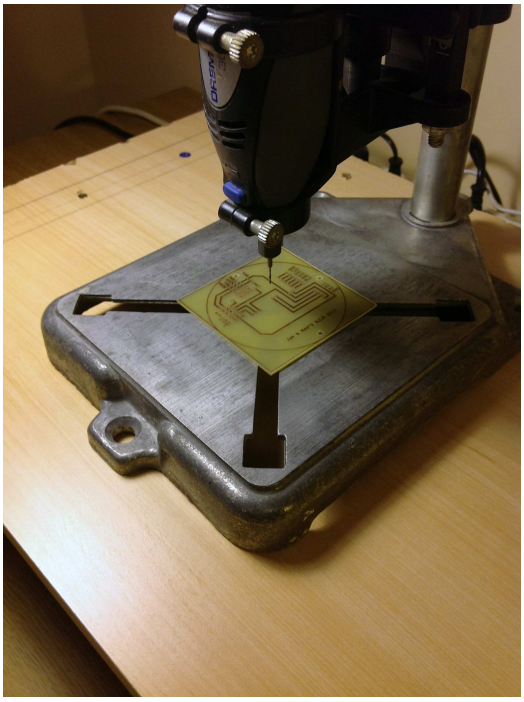


Figura 5.23: Processo di foratura

Il risultato ottenuto su cui poi abbiamo saldato tutte le nostre pin e componenti è stato il seguente:

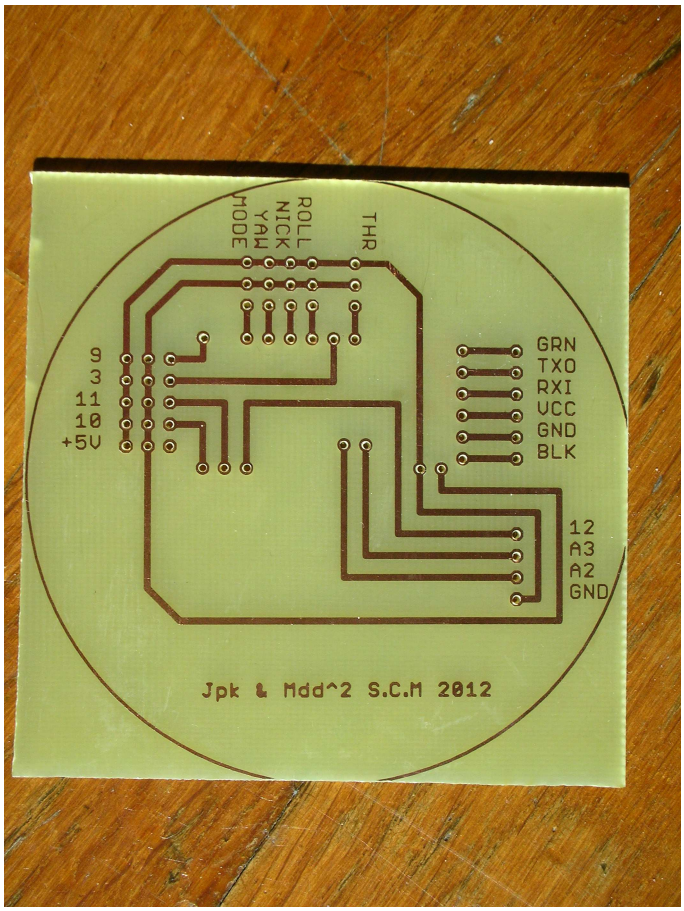
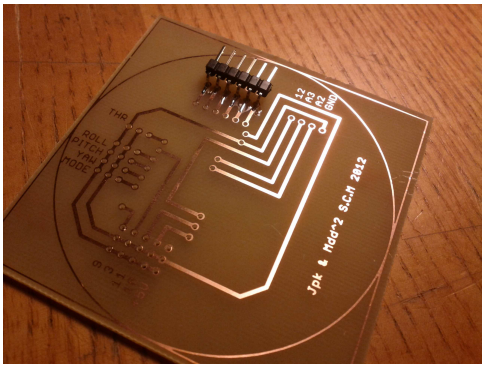
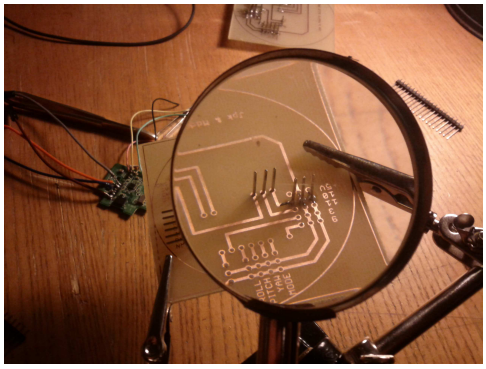


Figura 5.24: PCB finale. Da notare la particolare serigrafia

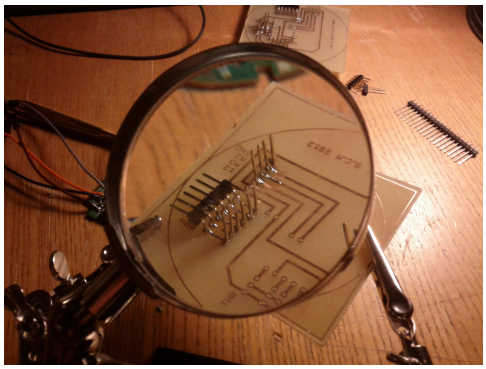
Dopo avere costruito la board siamo passati ad una lunga fase di saldatura della piedinatura. I pin saldati sono stati ben 62!



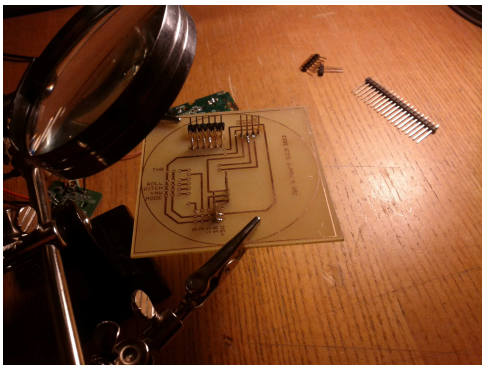
(a) Primissima fase di saldatura sulla appena costruita Board



(b) Le saldature proseguono



(c) L'uso della cosi detta 3^a mano ci è stata di grande aiuto



(d) Ancora saldature

Il risultato finale è stato complessivamente soddisfacente considerando che abbiamo provato la board con un apposito tester prima di installare la MCU ritratta in figura.

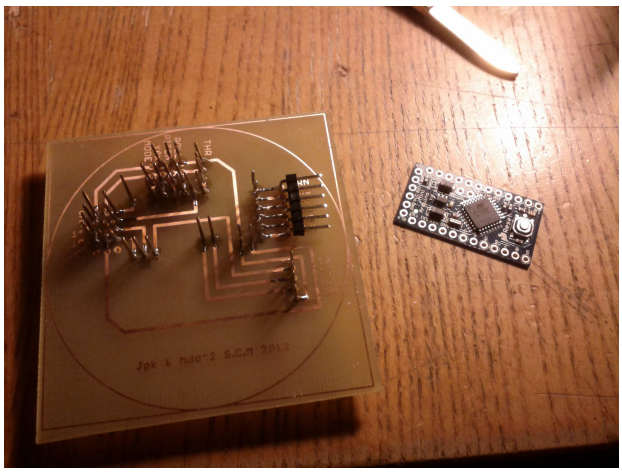


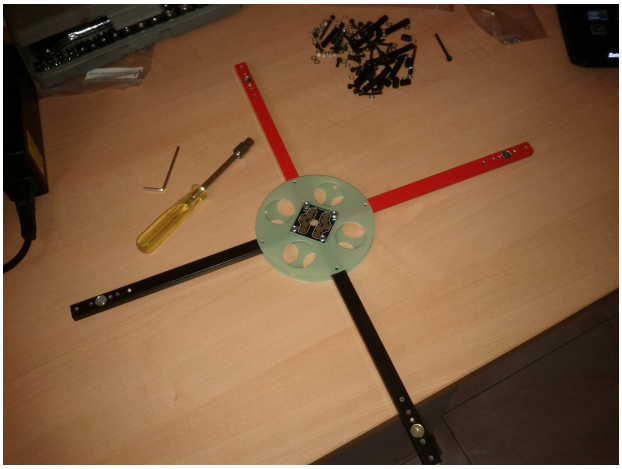
Figura 5.25: Board con saldature completate

5.2.2 Assemblaggio del frame

L'assemblaggio del frame ha richiesto un po di manualità ed è risultato essere semplice. I quattro bracci si concentrato verso una struttura centrale dove su più livelli è concentrata: l'elettronica di controllo comprensiva dei sensori appositamente posizionati, la batteria LIPO, e le 4 E.S.C per i motori. Alle estremità dei bracci sono installati i motori con delle strutture

atte a proteggerli nel malaugurato caso di avaria dei velivolo.
La costruzione è quindi iniziata con un struttura base costruita con i 4 bracci ed dei sostegni in vetroresina G10 per tenere il tutto assieme. Le seguenti immagini sono esplicative.

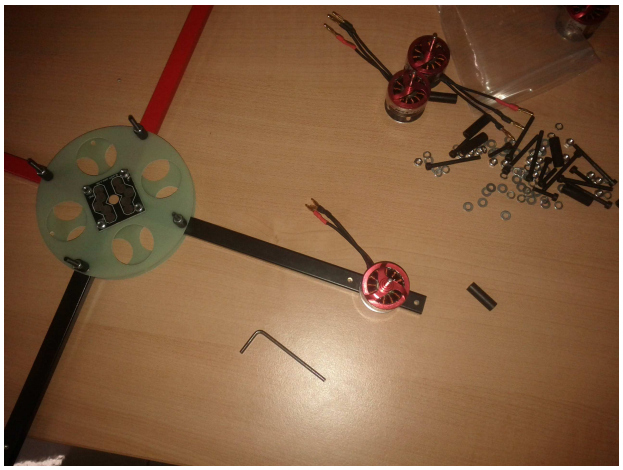
(a) I quattro bracci del telaio prima di essere assemblati



(b) I quattro bracci del telaio assemblati tramite la struttura centrale

Si osservi la presenza al centro dell'assieme in vetroresina la presenza di una PCB. Questo semplicissimo circuito per distribuire alle 4 E.S.C. la linea di alimentazione della batteria a 11.1V. In sostanza questo PCB non fa altro che mettere in comune i poli positivi e negativi sia delle 4 E.S.C. che quelli della batteria. Schematicamente ci si può riferire alla Fig. 5.1.
Dopo la costruzione di questa parte del frame siamo passati ad installare i motori alle estremità dei bracci. Questo è stato fatto con l'uso di due viti per motore.

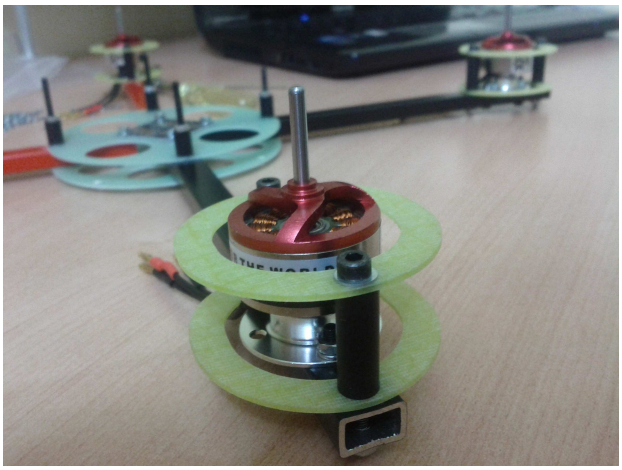
(a) Inizio installazione motori



(b) Installazione di un motore

(c) Si osservi sulla sinistra la protezione per il motore ancora non installata

Dopo avere installato i motori abbiamo installato le relative protezioni su ogni singolo motore, come si evince dalle Fig. **??** e Fig. **??**.



(a) Installazione delle protezioni per un motore

(b) Installazione delle protezioni su tutti i motori

Installati i motori abbiamo installato altri livelli della parte centrale del frame. In particolare le due immagini successive mostrano l'installazione del livello in cui va inserita la batteria LIPO.

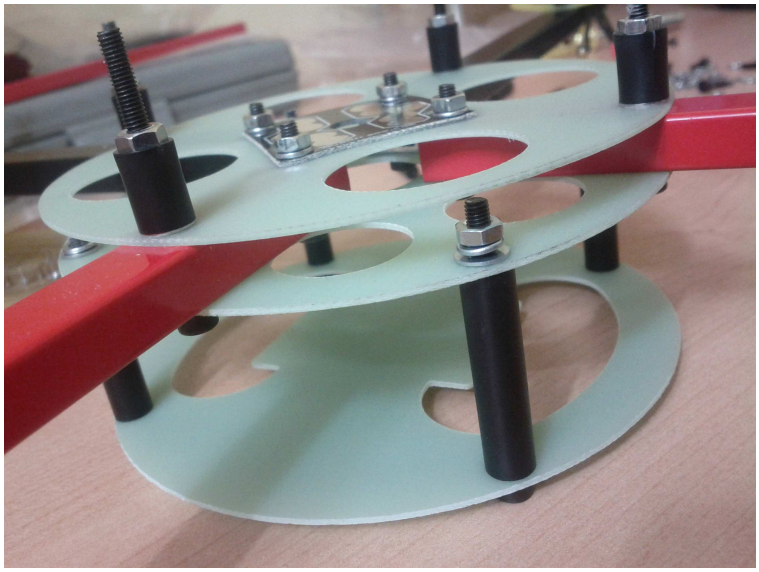


Figura 5.26: Particolare della torretta centrale del frame

Figura 5.27: Il frame dopo l'installazione del vano inferiore

Successivamente abbiamo passato i cavi di alimentazione facendoli passare per il centro del PCB per la distribuzione dell'alimentazione.

Figura 5.28: Passaggio dei cavi verso il vano batteria

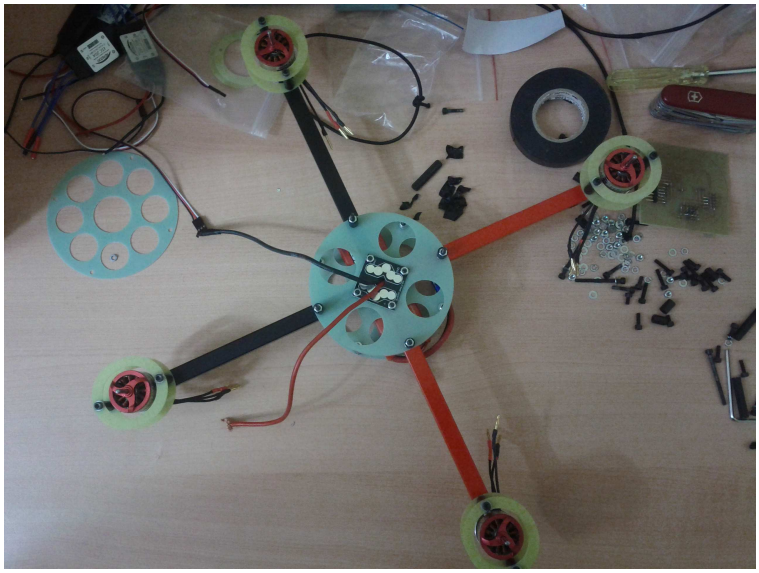


Figura 5.29: Passaggio dei cavi verso il vano batteria

Abbiamo poi posizionato le E.S.C e le abbiamo connesse ai motori ottenendo cosi il quadrirotore praticamente completo:

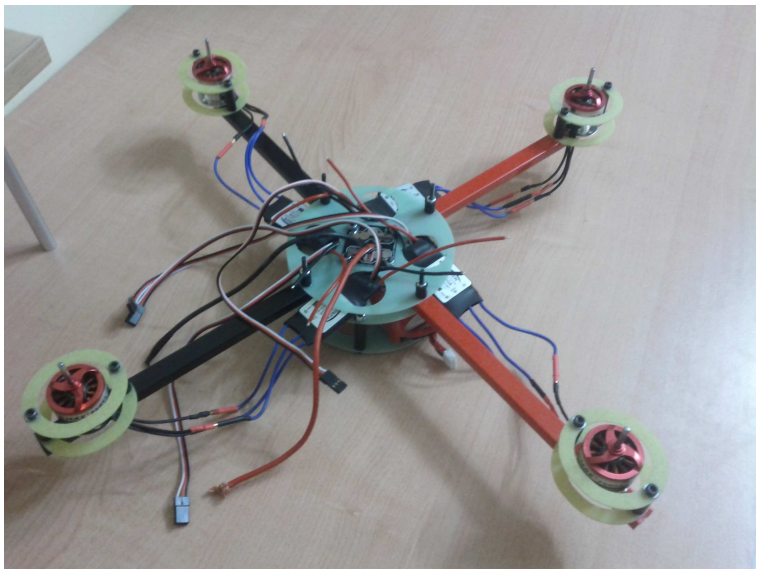


Figura 5.30: Passaggio dei cavi verso il vano batteria

L'ultimo passaggio prima di mostrare come è stata installa l'elettronica di controllo, e di trasmissione radio è stato quello dell'installazione delle eliche come in Fig.5.31

Figura 5.31: Quadcopter con eliche installate e senza elettronica di controllo

5.3 Software di controllo

Il controllore implementato è stato il LQ. Il software di controllo che abbiamo scritto è fa largo utilizzo delle librerie dibasso livello del ben noto progetto *Multiwii* la parte sostituita dopo n lunga lettura è quella relativa al controllo. A questo proposito si allega il file che abbiamo scritto.

```
1
2  #include <Wire.h> //Libreria per il basso livello a proposito della comunicazione con $I^2C$
3  #include <MegaServo.h>
4  #define NUM_SERVO 4
5  #define SERVO1_PIN 6
6  #define SERVO2_PIN 9
7  #define SERVO3_PIN 10
8  #define SERVO4_PIN 11
9  MegaServo Servos[NUM_SERVOS] ; //
10 //Variabili dall'accelerometro
11 float ax,ay,az; //Accelerazione in m/s^2
12 const float as=0.00488; //Risoluzione arduino
13
14 float psirad; // yaw in radians dal giroscopio
15 float phirad; // (rad)
16 float thetarad; // (rad)
17
18 // Impostazione filtro di Kalman
19 const float A[6][6] = {
20 {1.0000,0.0000,0.0000,−0.5985,−0.0000,0.0000},
21 {0.0000,1.0000,−0.0000,−0.0000,−0.5985,0.0000},
22 {−0.0000,−0.0000,1.0000,−0.0000,−0.0000,−0.0479},
23 95
24 {0.0500,0.0000,0.0000,0.3384,−0.0000,−0.0000},
25 {0.0000,0.0500,−0.0000,−0.0000,0.3384,0.0000},
26 {−0.0000,0.0000,0.0500,0.0000,0.0000,0.9147}
27 };//Matrice Ak di Kalman
```

```
28
29 // viene utilizzato il codice Imu_kalman per il filtraggio alla Kalman. Le matrici sono definite nel file def.h
30
31 float x[6] = {0,0,0,0,0,0}; //Vettore di stato: Roll speed, Pitch Speed, Yaw speed, Roll angle, Pitch angle, Yaw angle
32
33 // Dal basso livello
34
35 float u[10] = {0,0,0,0,0,0,0,0}; //vettore input: w1,w2,w3,w4,phi,theta,yaw la prima parte è il throttle
36
37 //Guadagni LQR in matrice
38 const float K[4][6] = {
39 {0.0000,7.1030,20.4163,0.0000,18.2839,14.4659},
40 {−7.1030,0.0000,−20.4163,−18.2839,0.0000,−14.4659},
41 {0.0000,−7.1030,20.4163,0.0000,−18.2839,14.4659},
42 {7.1030,0.0000,−20.4163,18.2839,0.0000,−14.4659}
43 };
44
45 float w1,w2,w3,w4; //
46 float w0; //Dal radiocomando
47
48 // Espressione di controllo del segnale di controllo e relative variabili
49 // Espressione di controllo del segnale di controllo e relative variabili
50 float U1=0;
51 float U2=0;
52 float U3=0;
53 float U4=0;
54 float wc1=0;
55 float wc2=0;
56 float wc3=0;
57 float wc4=0;
58
59 //Vettore di riferimento
60
61 float yrif[6]={0,0,0,0,0,0};
62 float xest[6]; // Stato stimato
63
64 void_init_lq(){
65 //scrive tutto a zero e viene chiamata dal main
66 motor[0].attach( SERVO1_PIN, 800, 2200); //Motore 1 − Nord− senso orario
67 motor[1].attach( SERVO2_PIN, 800, 2200); //Motore 2 − Est − senso anti orario
68 motor[2].attach( SERVO3_PIN, 800, 2200); //Motore 3 − Sud − senso orario
69 motor[3].attach( SERVO4_PIN, 800, 2200); //Motore 4 − Ovest − senso antiorario
70
71 //Prima di tutto ferma i motori
72
73 motor[0].write(0);
74 motor[1].write(0);
75 motor[2].write(0);
76 motor[3].write(0);
77
78 }
79
80 void processa_lettura(){
81
```

```
ax = get_data_acc[x]; //Acc x
ay = get_data_acc[y]; //Acc x
az = get_data_acc[z];//Acc z

phirad=get_data_gyro[yaw];
pitchrad=get_data_gyro[pitch];
phirad=get_data_gyro[roll];
}


void azione_controllo(){
xest=get_filtered(Ak,phirad,thetarad,psirad,ax,ay,az);

U1=K[0][0]*(xest[0]-yrif[0])+K[0][1]*(xest[1]-yrif[1])+K[0][2]*(xest[2]-yrif[2])+K[0][3]*(xest[3]-yrif[3])+K[0][4]*(xest[4]-yrif[4])+K[0][5
U2=K[1][0]*(xest[0]-yrif[0])+K[1][1]*(xest[1]-yrif[1])+K[1][2]*(xest[2]-yrif[2])+K[1][3]*(xest[3]-yrif[3])+K[1][4]*(xest[4]-yrif[4])+K[1][5
U3=K[2][0]*(xest[0]-yrif[0])+K[2][1]*(xest[1]-yrif[1])+K[2][2]*(xest[2]-yrif[2])+K[2][3]*(xest[3]-yrif[3])+K[2][4]*(xest[4]-yrif[4])+K[2][5
U4=K[3][0]*(xest[0]-yrif[0])+K[3][1]*(xest[1]-yrif[1])+K[3][2]*(xest[2]-yrif[2])+K[3][3]*(xest[3]-yrif[3])+K[3][4]*(xest[4]-yrif[4])+K[3][5

w1=-0.2500*U1+0.5000*U2+U3*0.2500; // calcolo uscite perché il controllore ragione su differenze di velocità
w2=-0.2500*U2+0.5000*U3-0.2500*U4;
w3=-0.2500*U2-0.5000*U3+0.2500*U4;
w4=-0.2500*U1-0.5000*U1-0.2500*U4;

wc1=w0-w1; // applico rispetto al riferimento
wc2=w0-w2;
wc3=w0-w3;
wc4=w0-w4;
}

sig_to_mot_pwm(){
p1=(int)((wc1/2.0276)+1185); // (velocità motore / guadagno) + zona morta
p2=(int)((wc2/1.8693)+1262);
p3=(int)((wc3/2.0018)+1255);
p4=(int)((wc4/1.9986)+1255);
}
```

Bibliografia

[1] G. Celentano e L. Celentano. *Fondamenti di Dinamica dei sistemi, Vol. II.* Edises, Campania, Napoli, 1 edition, 2010.

[2] InvenSense. Produttore Giroscopio. `www.invensense.com`, 2012. [Online; ultimo accesso 31-03-2012].

[3] Jackub Jeuwla. Telaio. `http://quadframe.com/collections/multicopter-frames/products/quad001`, 2010. [Online; ultimo accesso 12-09-2011].

[4] Jackub Jeuwla. Costruttore telaio. `http://quadframe.com/`, 2011. [Online; ultimo accesso 12-01-2012].

[5] Alfred Price. *Instruments of Darkness: The History of Electronic Warfare, 1939-1945.* Greenhill Books, 3 edition, 2006.

[6] Eagle Cad Software. Software disegno PCB. `http://www.eaglecad.com/`, 2011. [Online; ultimo accesso 28-11-2011].

[7] Wikipedia. Inter Integrated Circuit. `http://it.wikipedia.org/wiki/I%C2%B2C`, 2012. [Online; ultimo accesso 31-03-2012].

[8] Wikipedia. Sensori MEMS. `http://it.wikipedia.org/wiki/MEMS`, 2012. [Online; ultimo accesso 31-03-2012].

Elenco delle figure

Elenco delle tabelle